

CHoRaL: Collecting Humor Reaction Labels from Millions of Social Media Users

EMNLP 2021 Best Short Paper

인공지능 연합전공
2021-27633 황예린

ABSTRACT

- Desire to understand user-generated content.
- But difficult to collecting a large scare humor dataset with reliable humor labels.

CHoRaL

- A framework to generate perceived humor labels on Facebook posts.
- Using the naturally available user reactions.
- Able to make both binary labels & continuous scores of humor and non-humor.

INTRODUCTION

- Most attempts have focused on distinguishing between jokes / news.
- But far from real-world scenarios where humor and non-humor come from same domain.
- **CHoRaL**: a framework for **C**ollecting **H**umor **R**eaction **L**abels

Got kicked out from the hospital because I told the Covid patients to stay positive.	😂 295	👍 3	
Too many families have lost a loved one to COVID-19. This is a disease that doesn't discriminate — we need to come together & meet this moment.	👍 253	❤️ 39	😭 34
	😂 10	😬 7	😡 4
	😱 1		

1. Labeling humor on any Facebook post.
2. Providing both binary & continuous labels.
3. Enabling the collection of large-scale social media datasets on humor.

CHoRaL Framework

- Data collection & cleaning
- Facebook posts from CrowdTangle by searching COVID-related keywords.
- Text-only post, set language as English.

- Humor Score (HS)
$$\text{HS} = \frac{h}{t} * \tanh\left(\frac{t}{50}\right)$$

- Non-humor Score (NS)
$$\text{NS} = -\log\left(\tanh\left(\frac{t}{50}\right)\right) * \sum_{r \in R} \frac{(S(r) - O(r))^2}{|R|}$$



785k posts

# of Posts	784,965
# of Poster Accounts	264,685
# of User Reactions	126,839,984
# of Haha Reactions	6,525,247

Humor Detection Experiments

- 40k post 80:20 split
- Fine-tuned 3 pretrained language models
 1. RoBERT
 2. BERTweet
 3. BERTweet-covid

- Result

	Continuous		Binary	
	F1	AUC	F1	AUC
Human	-	-	0.867	-
RoBERTa	0.869	0.939	0.868	0.937
BERTweet	0.879	0.947	0.881	0.950
BERTweet-covid	0.880	0.948	0.883	0.951

Conclusion

- CHoRaL framework for automatically collecting humor reaction labels.
- Dataset including 785k posts with humor and non-humor scores.
- Build models to detect humor.
- CHoRaL enables the development of humor detection model on any topic.
- Also can be used to label other human reactions such as anger and sadness.

감사합니다



LiDAR Place recognition OverlapNet

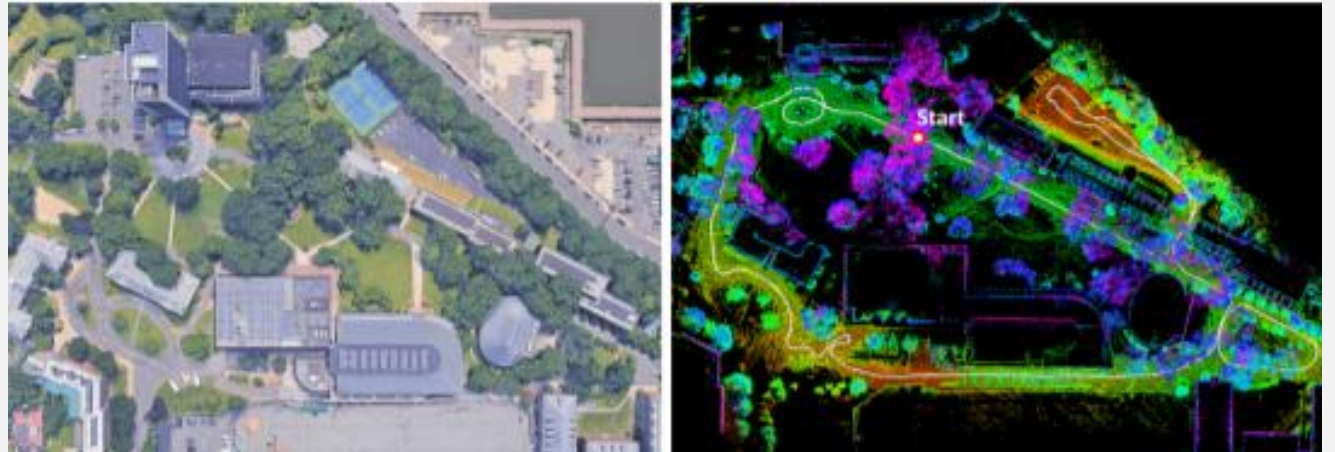
전기정보공학부
2021-23297
김동욱

Contents

- Problem Background
- Paper method, result

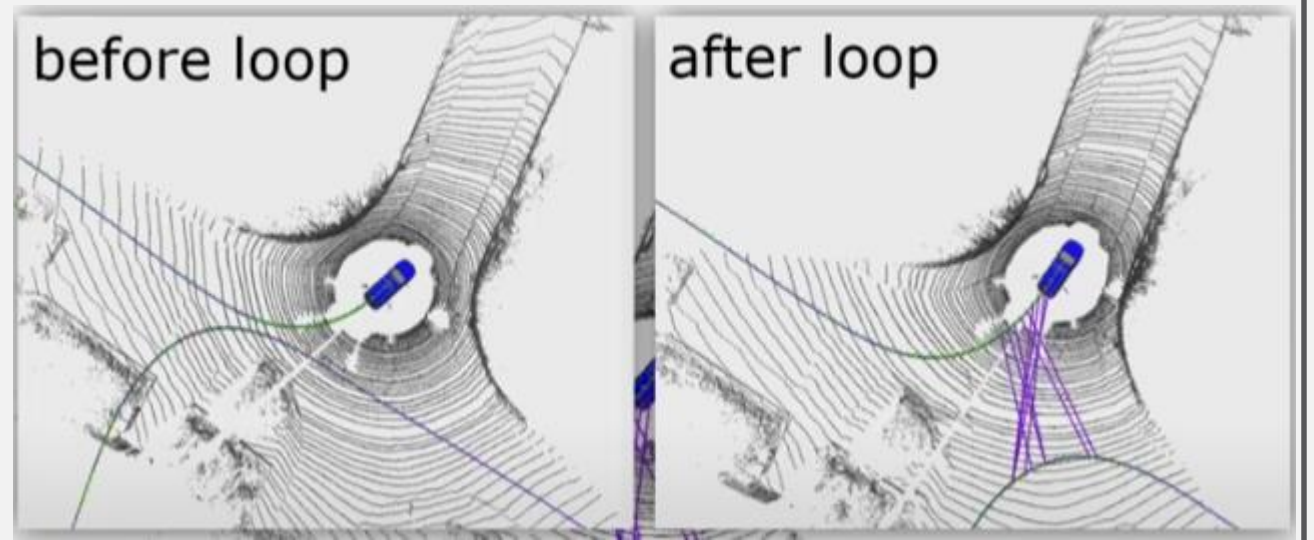
SLAM

- SLAM: Simultaneous Localization and Mapping
 - 현재 자신의 위치(Localization)와 주변 사물의 위치(Mapping)를 동시에 파악
 - 매 시간마다 연속된 frame을 이어붙이는 registration
- 시간이 지날수록 drift error accumulation
- 에러를 보정하기 위해 loop closure



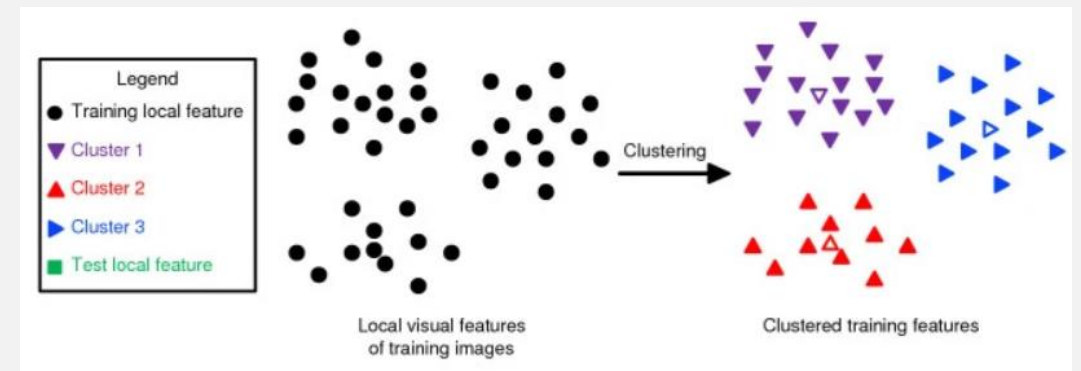
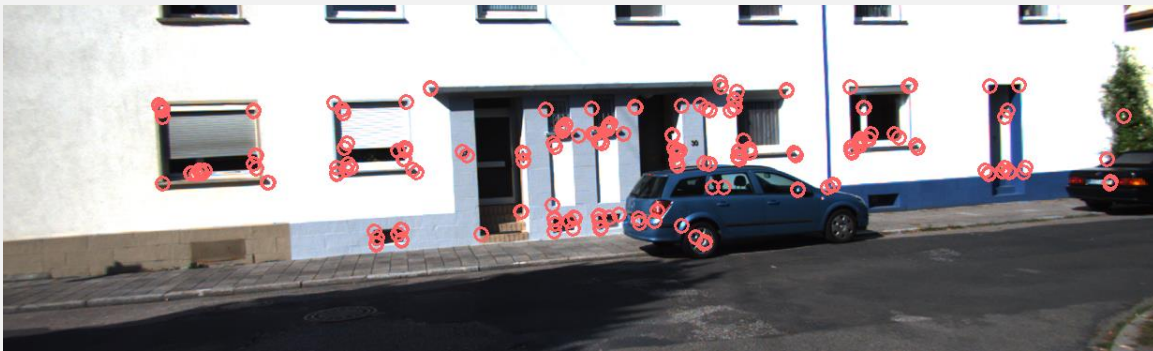
Loop closure

- 동일한 장소에 다시 돌아왔을 때 그동안 쌓인 error 보정
- 과거에 방문한적이 있는지 판단하는 place recognition 알고리즘 필요
- OverlapNet: Loop Closing for LiDAR-based SLAM
- Proceedings of Robotics: Science and System (RSS) 2020



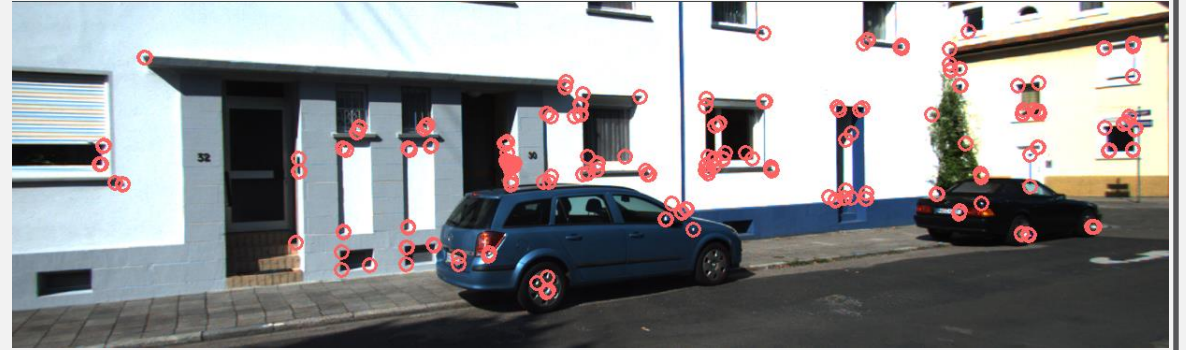
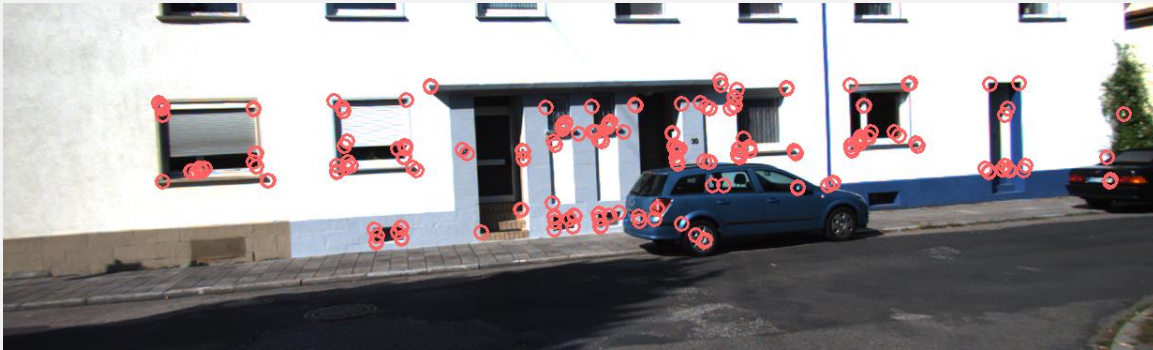
Place recognition

- 가장 Classic한 place recognition 알고리즘은 bag-of-words
 - 현재 image로부터 feature vector extraction
 - Feature들을 k-nearest-neighbor로 clustering하여 vocabulary set을 생성



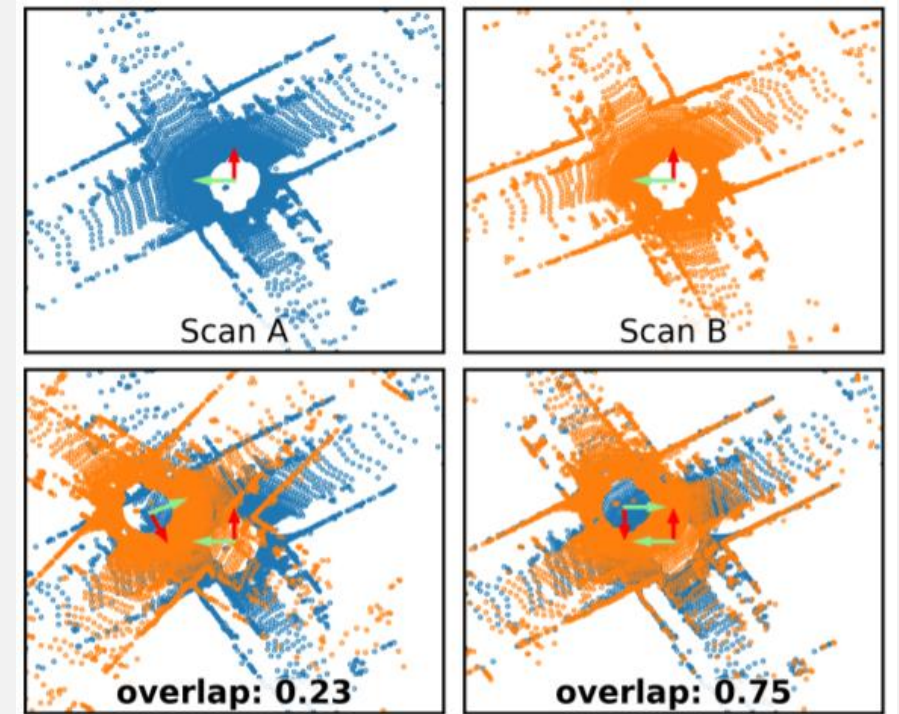
Place recognition

- 가장 Classic한 place recognition 알고리즘은 bag-of-words
 - 현재 image로부터 feature extraction
 - Feature들을 k-nearest-neighbor로 clustering하여 vocabulary set을 생성
 - 새로운 image가 들어왔을 때 기존에 들어왔던 image와 가지고있는 vocabulary가 비슷한지 확인
 - 일정 threshold 이상으로 유사하면 같은 장소로 판별



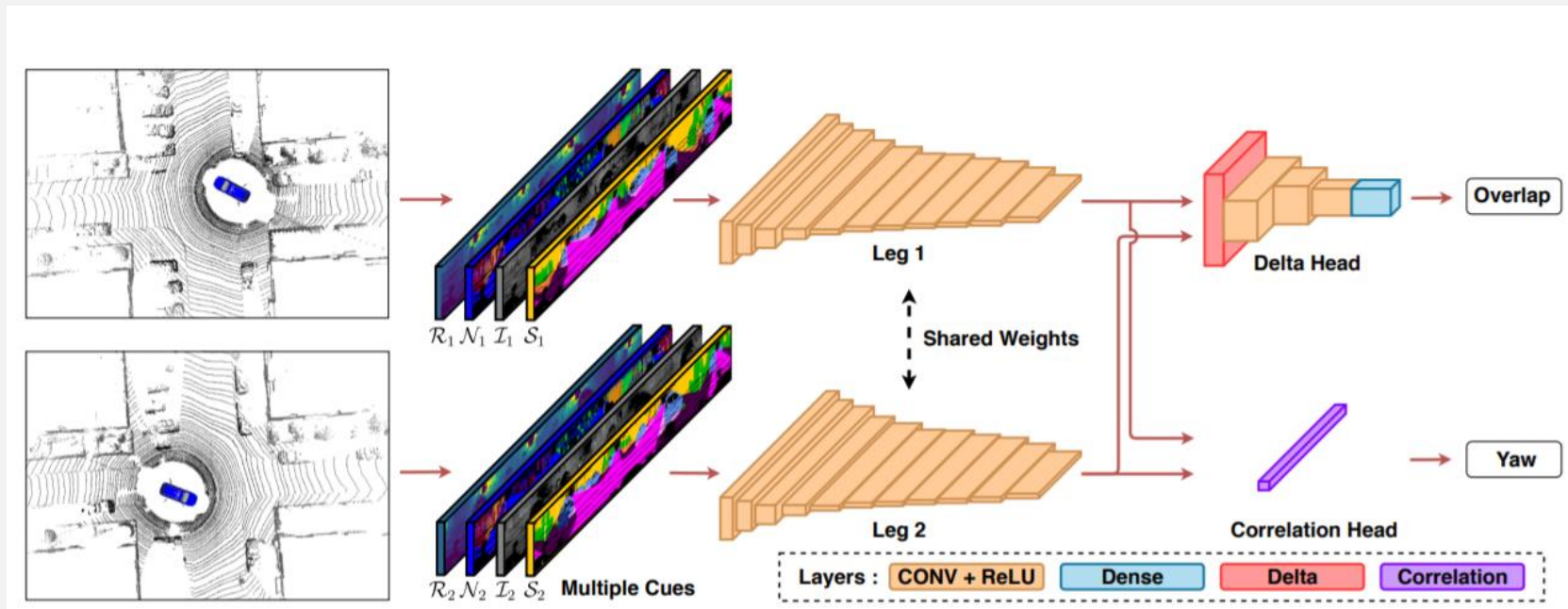
OverlapNet

- LiDAR pointcloud를 image형태로 변환
- end-to-end Neural network 방법을 사용하여 place recognition 진행
- 두 장소 사이의 overlap의 정도를 정의하여 training
- 두 장소 사이 relative rotation angle을 계산해냄



OverlapNet

- 두 input point cloud를 image 형태로 project
- 총 4가지 image를 siamese network에 input으로 사용
 - Range / Normal / Intensity / Sementic image



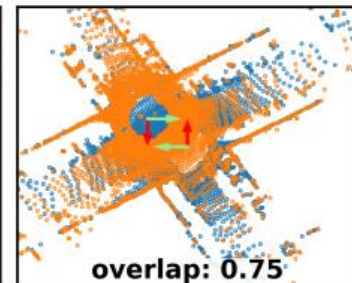
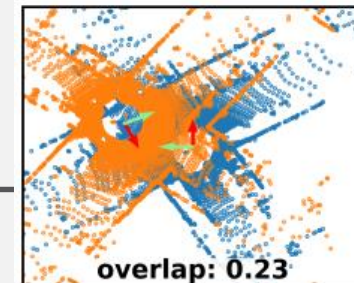
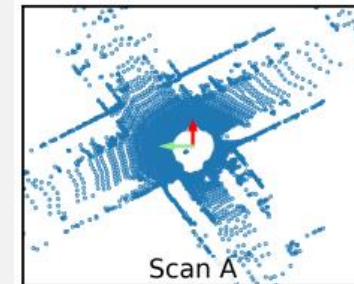
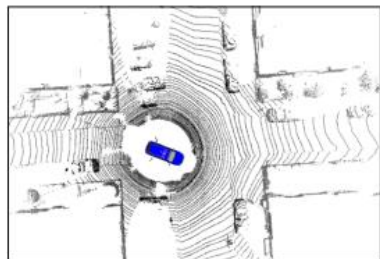
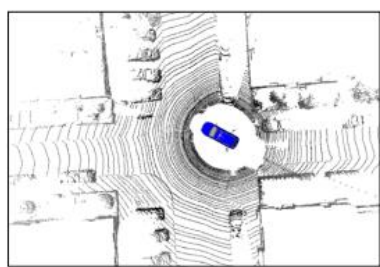
Overlap의 정의

- 두 개의 LiDAR scan의 pose(C_1, C_2)를 알고 있다면
- Scan1의 point를 Scan2의 coordinate로 transformation 후 range map 생성(V'_1)

$$\mathcal{V}'_1 = \Pi(\mathbf{T}_{WC_1}^{-1} \mathbf{T}_{WC_2} \mathcal{P}_1)$$

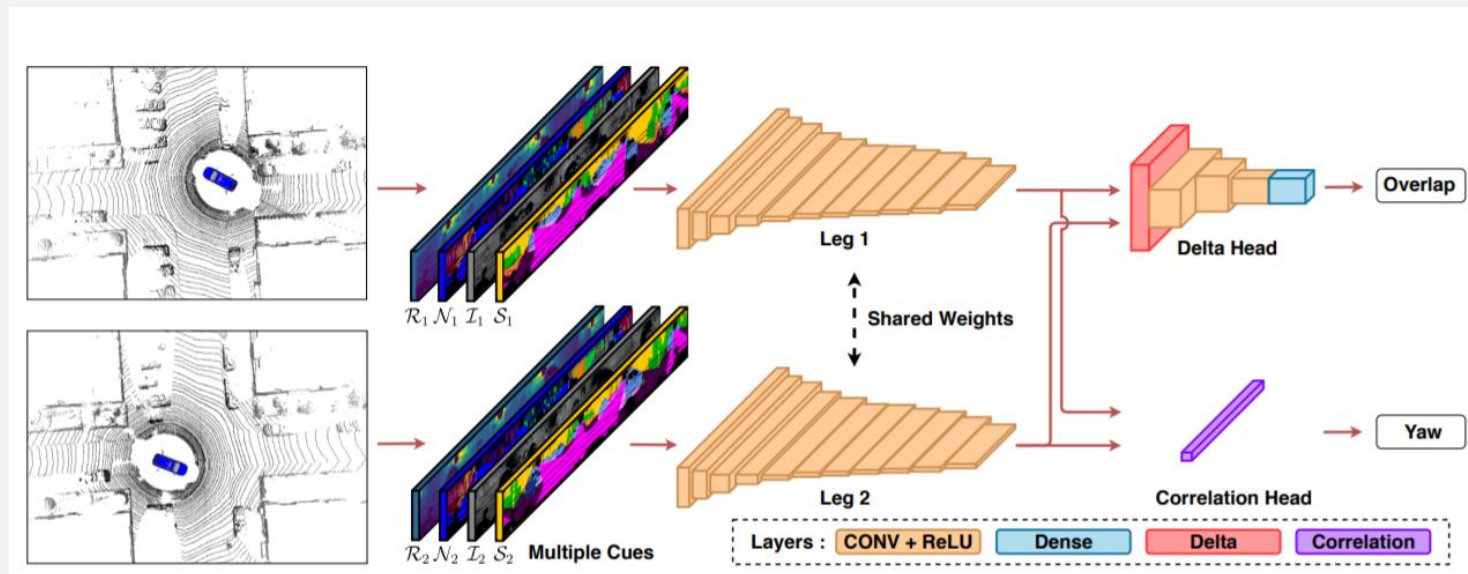
- Scan2의 range map V_2 과 Scan1로부터 생성된 V'_1 을 비교하여 Overlap 계산

$$O_{C_1 C_2} = \frac{\sum_{(u,v)} \mathbb{I}\{\|\mathcal{V}'_1(u,v) - \mathcal{V}_2(u,v)\| \leq \epsilon\}}{\min(\text{valid}(\mathcal{V}'_1), \text{valid}(\mathcal{V}_2))}$$

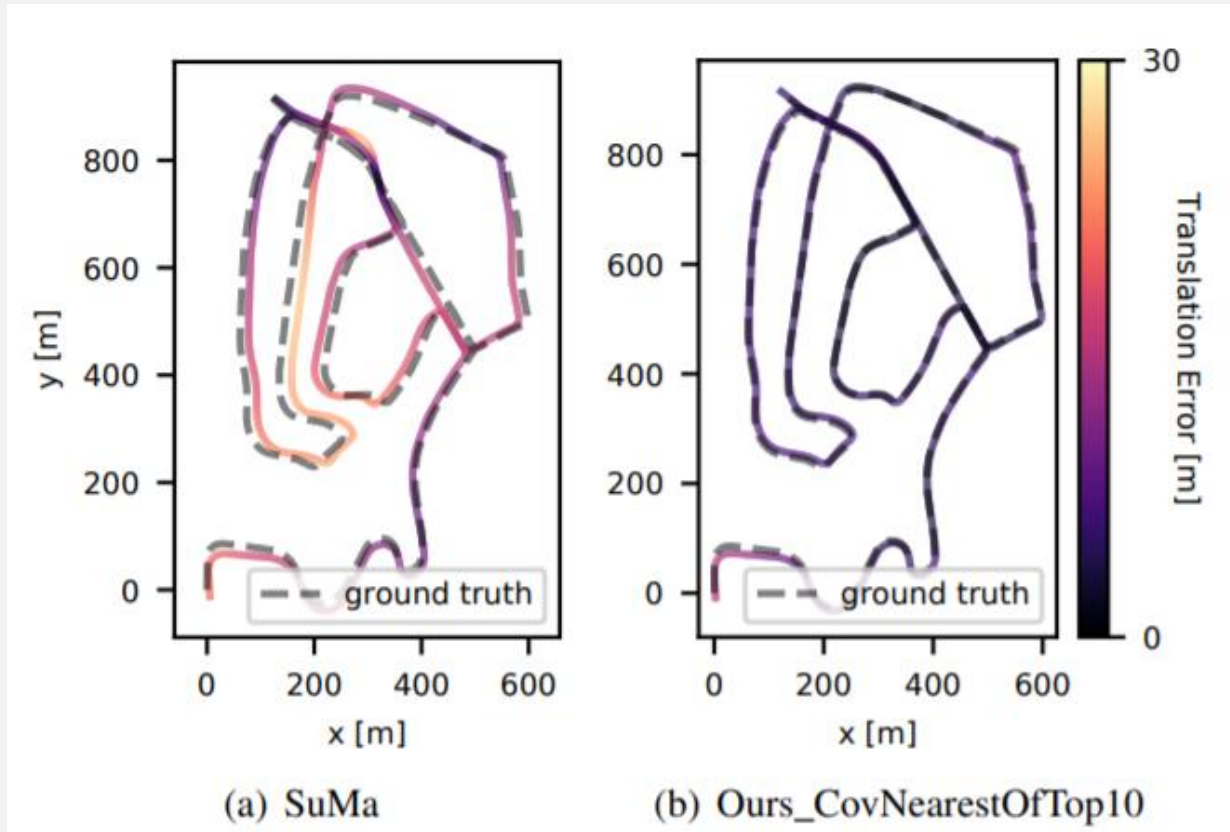


Overlap의 정의

- 실제 real world에서는 모든 장소의 pose를 알 수 없음
- 따라서 이 방식의 overlap 측정은 train set 생성시에 ground truth로 사용됨
- 실제 test 단계에서는 pose 정보 없이 overlap 측정



Result



- 올바른 Loop closure을 통해 accumulate error 보정 가능
- 보정 과정에서 필요한 relative transformation의 initial 값을 제공 가능

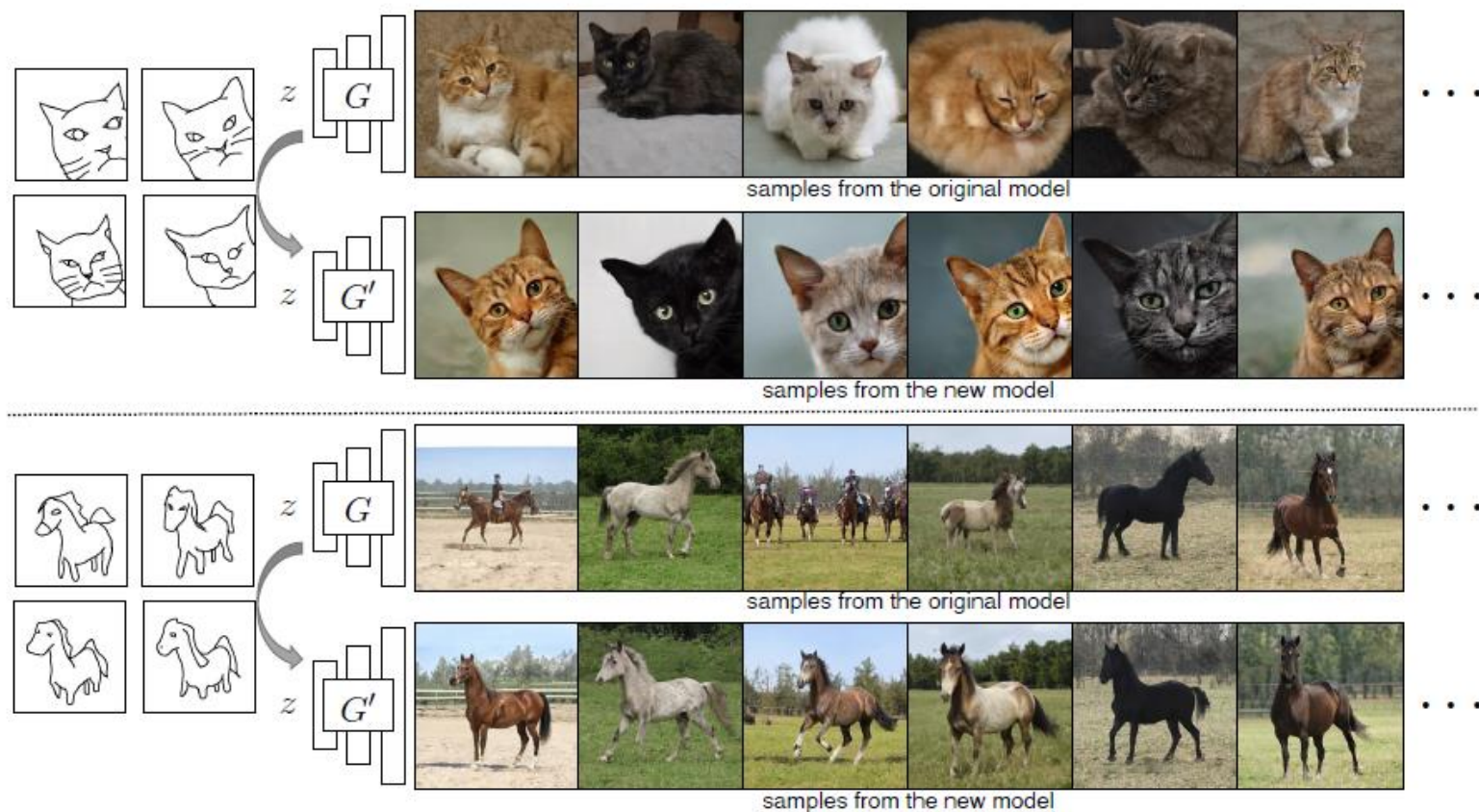
Thank you!

Sketch Your Own GAN

<ICCV 2021>

발표자 : 김민우

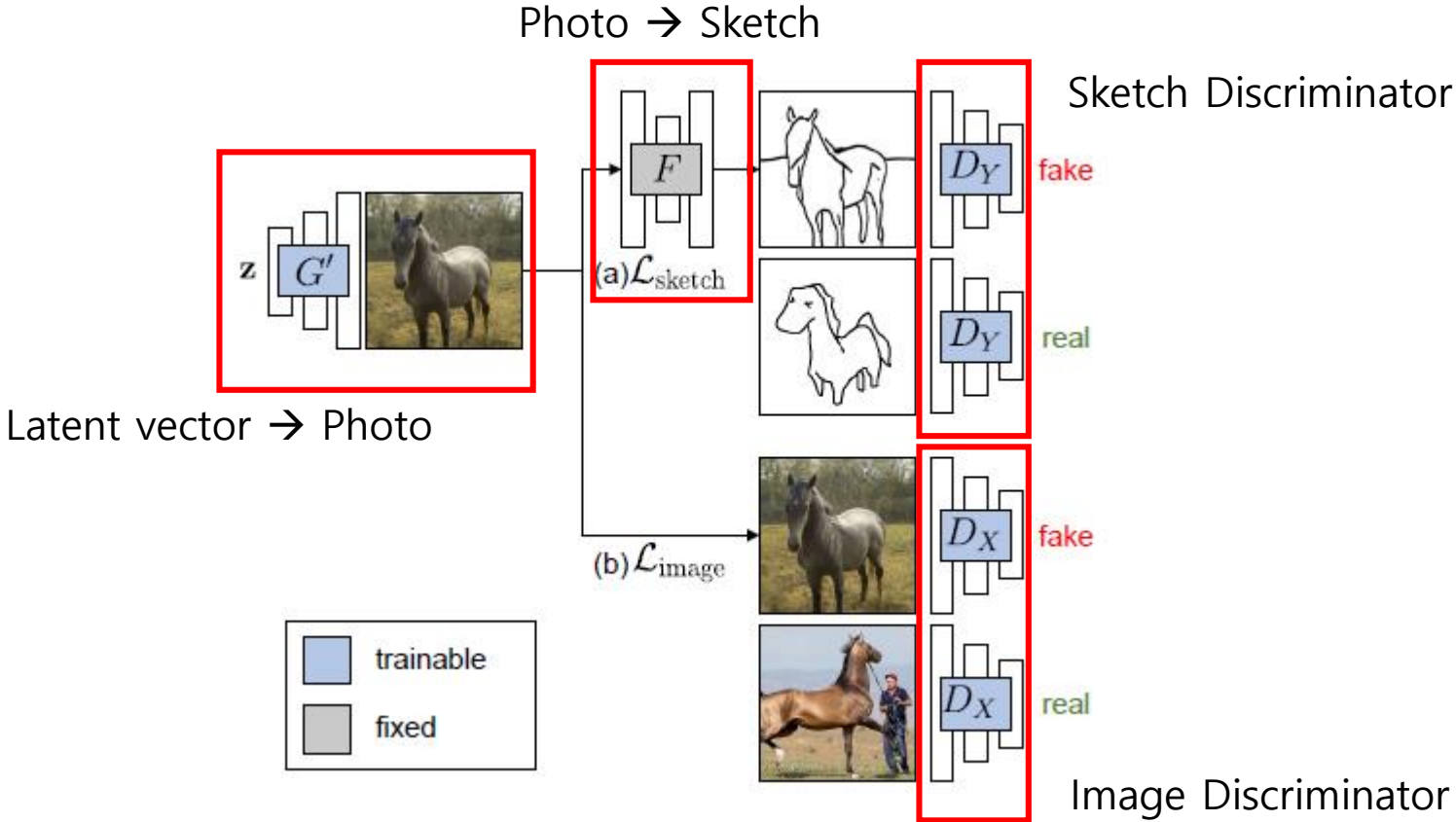
Task



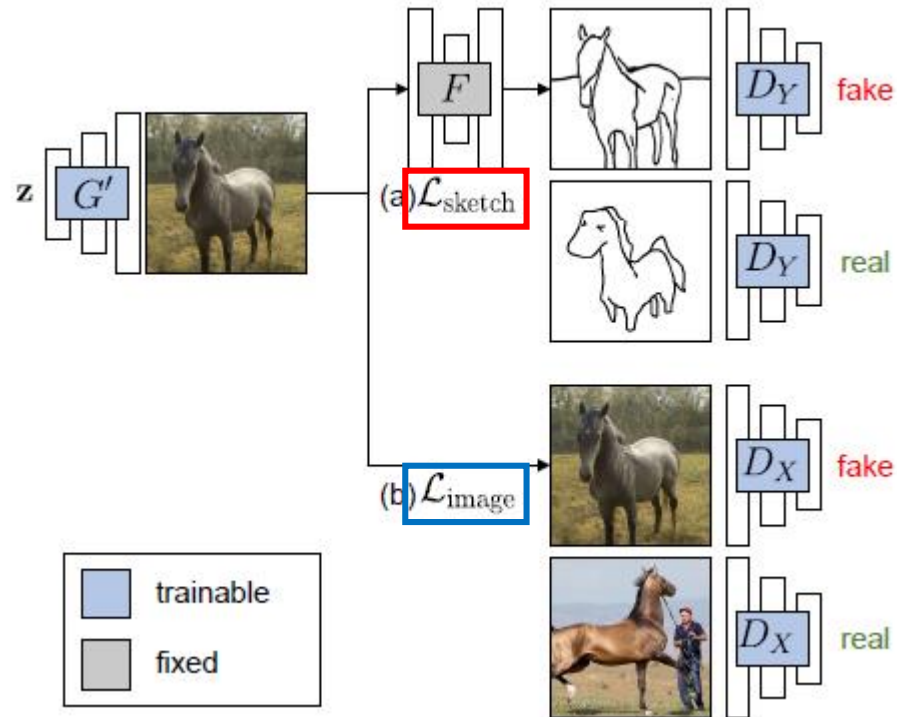
(a) User sketches

(b) Customizing a GAN using human sketches

Network Architecture

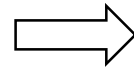


Objective Function



$$\mathcal{L}_{\text{sketch}} = \mathbb{E}_{\mathbf{y} \sim p_{\text{data}}(\mathbf{y})} \log(D_Y(\mathbf{y})) + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} \log(1 - D_Y(F(G(\mathbf{z}))))$$

$$\mathcal{L}_{\text{image}} = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} \log(D_X(\mathbf{x})) + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} \log(1 - D_X(G(\mathbf{z})))$$

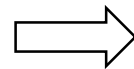


$$\mathcal{L} = \mathcal{L}_{\text{sketch}} + \lambda_{\text{image}} \mathcal{L}_{\text{image}}$$



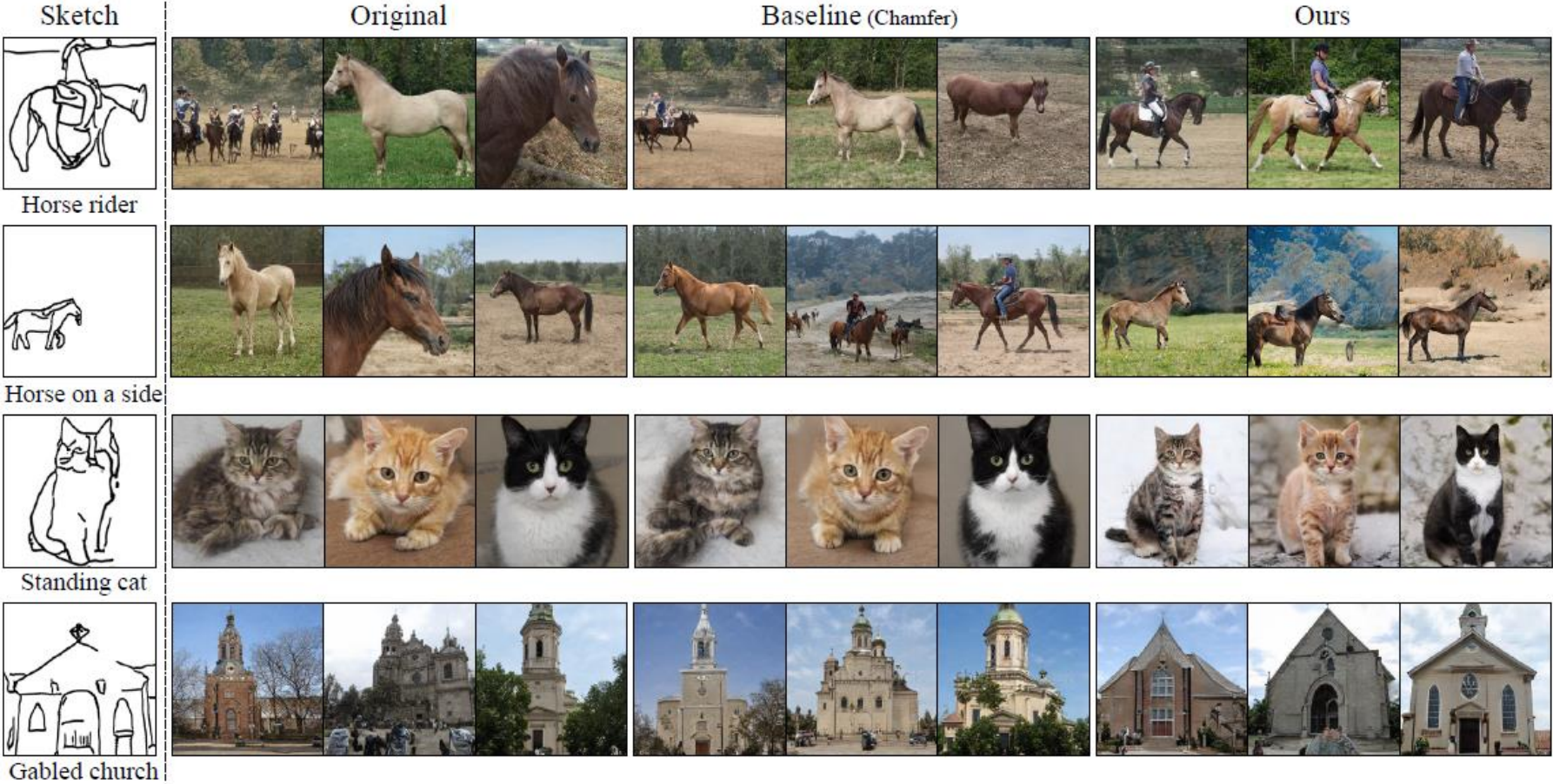
$$\mathcal{L}_{\text{weight}} = \|\theta' - \theta\|_1$$

Regularization term



$$\theta' = \arg \min_{\theta'} \max_{D_X, D_Y} \mathcal{L}$$

Experiment

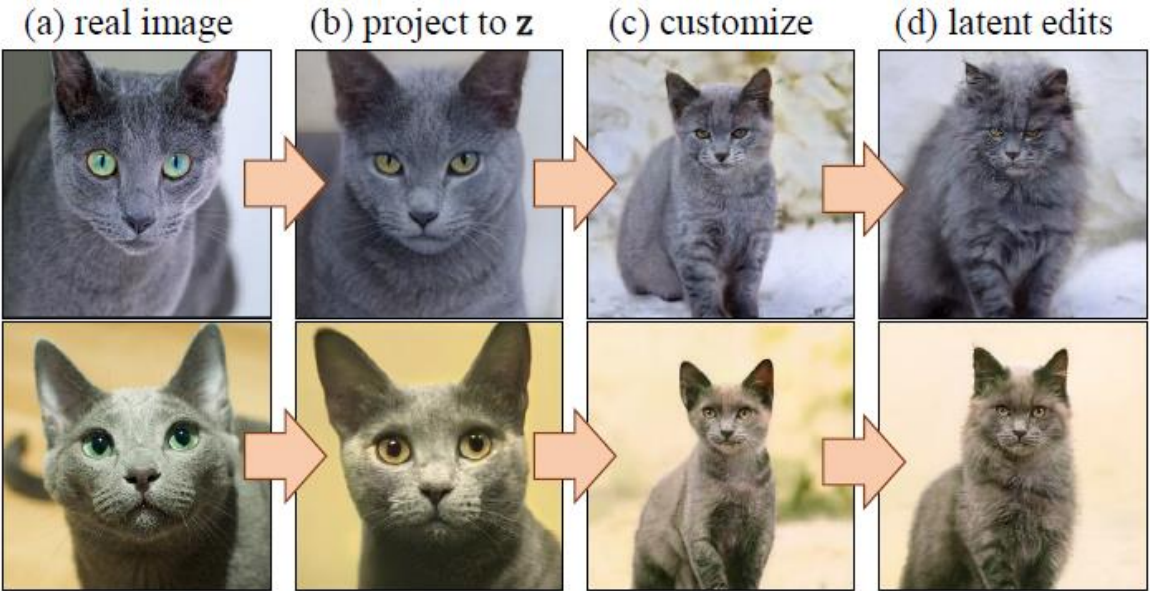


Experiment

Latent space interpolation



Image editing



Thank you

Online Knowledge Distillation via Collaborative Learning

Qiushan Guo, Xinjiang Wang, Yichao Wu, Zhipeng Yu, Ding Liang, Xiaolin Hu, Ping Luo;

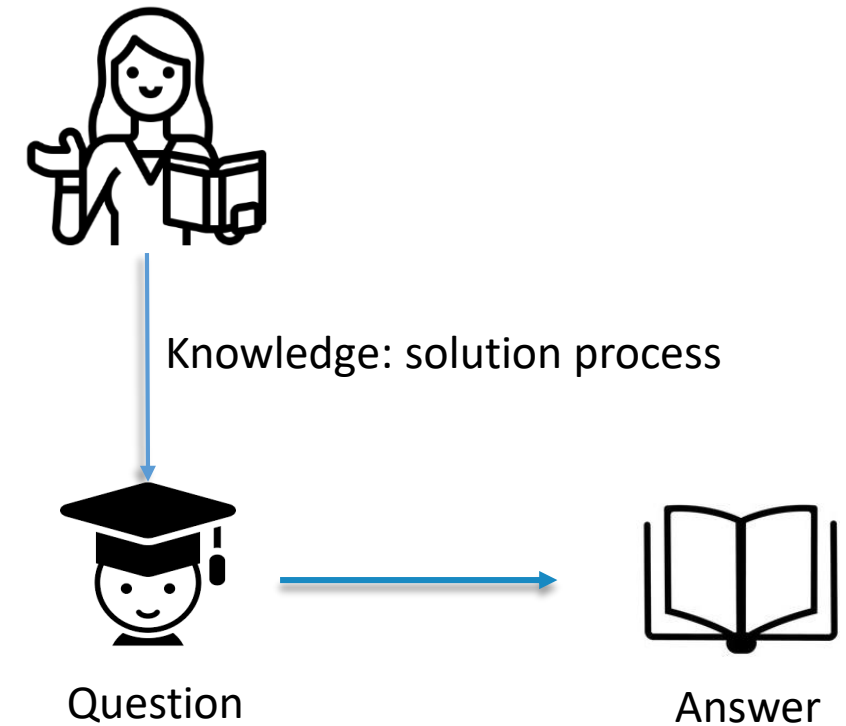
Guo, Qiushan, et al. "Online knowledge distillation via collaborative learning." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.



Introduction

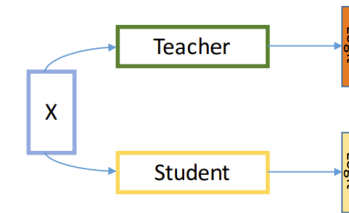
Background of Knowledge Distillation

- $L_{KD} = \frac{1}{n} \sum_{i=1}^n T^2 KL(\mathbf{p}_i, \mathbf{q}_i)$
 - n : batch size, T : temperature parameter.
 - p & q : soften probability distribution produced by teacher and student network.
 - p : $\text{softmax}\left(\frac{z_t}{T}\right)$, q : $\text{softmax}\left(\frac{z_s}{T}\right)$, where z_t & z_s represents the logit of teacher and student.

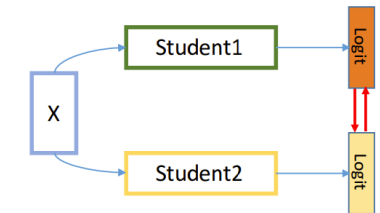


Introduction

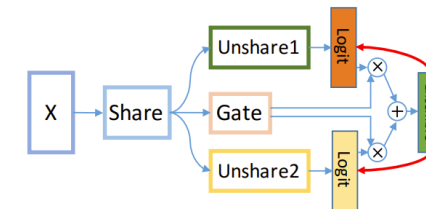
- Baseline: transfers knowledge from the static pre-trained teacher to student model.
- DML: Students can learn from each other.
- ONE: establishes teacher using multiple branch design.
- KDCL: consistently gains extra information from ensembling soft target produced by all students, outperforming existing approaches.
- Question: Could we use a small network to improve the model with larger capacity in a one-stage distillation framework?



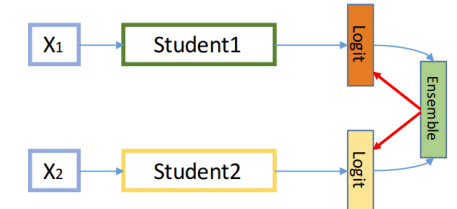
(a) Baseline



(b) DML



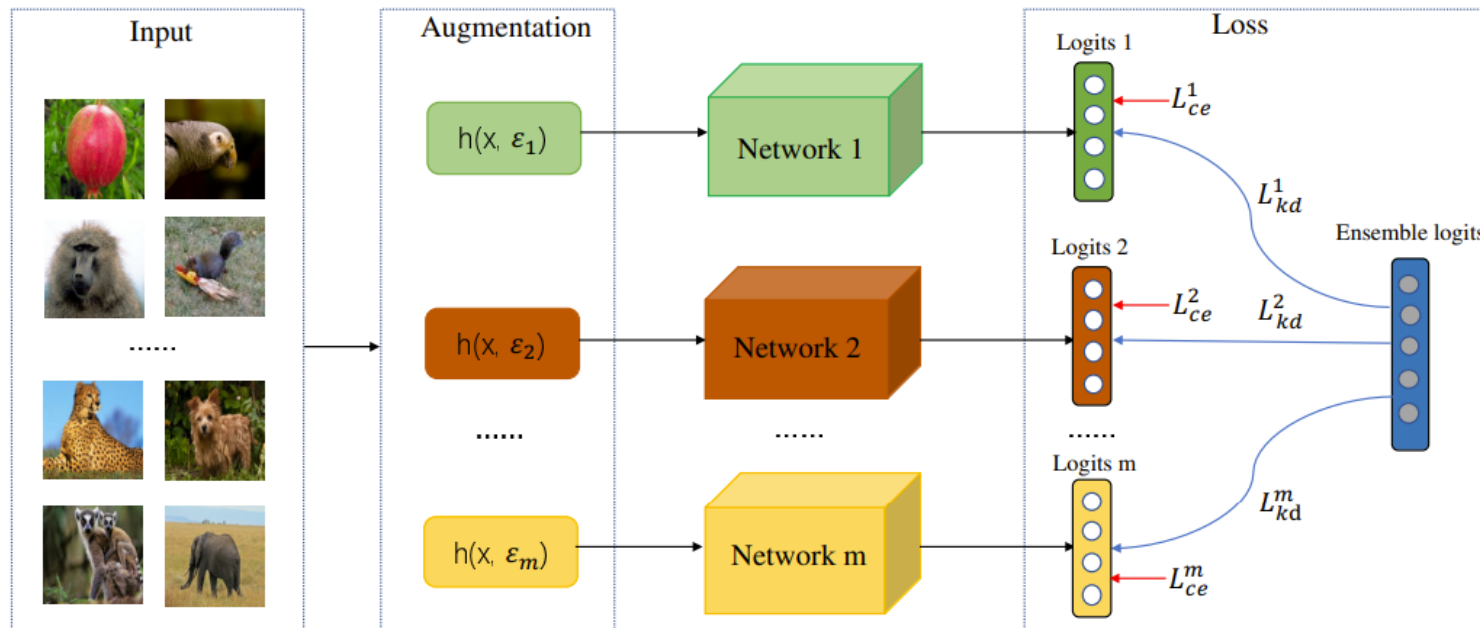
(c) ONE



(d) KDCL

Proposed method

- A new pipeline of knowledge distillation based on collaborative learning is designed. Models of various learning capacity can benefit from collaborative training.
- A series of model ensembling methods are designed to dynamically generate high-quality soft targets in a one-stage online knowledge distillation framework.



Proposed method

■ Loss function

- $$L = \sum_{i=1}^m L_{CE}^i + \lambda L_{KD}^i$$

- CE: cross-entropy loss, KD: knowledge distillation loss (KL divergence between the output of students and the soft target, λ : trade-off weight.

■ KDCL-Naive

- $\mathbf{z}_t = h(\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_m)$
- h : function to produce higher quality logit compared with the logits of students.
- $\mathbf{z}_t = \mathbf{z}_k, k = \arg \min_i L_{CE}(\mathbf{z}_i, \mathbf{y})$
- \mathbf{y} : one-hot label, L_{CE} : standard cross-entropy loss

Proposed method

■ KDCL-Linear

- Defines the teacher logit as the best linear combination of sub-network logits.
- $\mathbf{Z} = (\mathbf{z}_1^T; \mathbf{z}_2^T; \dots; \mathbf{z}_m^T)$, each column of the matrix represents the logit of a student.
- Problem: $\min_{\alpha \in \mathbb{R}^m} L_{CE}(\alpha^T \mathbf{Z}, \mathbf{y})$, subject to $\sum_{i=1}^m \alpha_i = 1, \alpha_i \geq 0$

■ KDCL-MinLogit

- $\mathbf{p} = \text{softmax}(\mathbf{z}) = \text{softmax}(\mathbf{z} - z^c)$
- z^c is the element corresponding to target class c in logit.
- $z_{t,j} = \min\{Z_{j,i}^c | i = 1, 2, \dots, m\}$

Proposed method

■ KDCL-General

- $E(\mathbf{x}) = (f(\mathbf{x}) - t)^2$

- $$E = \int \left(\sum_{i=1}^m w_i f_i(\mathbf{x}) - t \right)^2 p(\mathbf{x}) d\mathbf{x}$$

$$= \sum_{i=1}^m \sum_{j=1}^m w_i w_j C_{ij},$$

- Where $p(x)$ is the data distribution and C_{ij} is expressed as
$$C_{ij} = \int (f_i(\mathbf{x}) - t)(f_j(\mathbf{x}) - t) p(\mathbf{x}) d\mathbf{x}$$

- $$w_k = \frac{\sum_{j=1}^m C_{kj}^{-1}}{\sum_{i=1}^m \sum_{j=1}^m C_{ij}^{-1}}$$

$$\approx \frac{1}{N} \sum_{k=1}^N (f_i(\mathbf{x}_k) - t)(f_j(\mathbf{x}_k) - t)$$

- where w_k is the k-th element of the optimum weight w and C_{ij}^{-1} is the value of the i -th row and the j -th column of the inverse matrix of C .

Experiments

■ Results on ImageNet

- 1000 object classes with 1.28 million images
- Learning rate starts at 0.1 and warms up to 0.8 linearly after 5 epochs.
- Weight decay: 0.0001, batch size: 2048, momentum:0.9.

Method	ResNet-50	ResNet-18	Gain
Vanilla	76.8	71.2	0
KD[10]	76.8	72.1	0.9
DML[32]	75.8	71.7	-0.5
ONE[15]	-	72.2	-
CLNN[22]	-	72.4	-
KDCL-Naive	77.5	72.9	2.4
KDCL-Linear	77.8	73.1	2.9
KDCL-MinLogit	77.8	73.1	2.9
KDCL-General	77.1	72.0	1.1

Conclusion

- With models of ensemble method, models of various capacity can benefit from collaborative learning.
- Different ensemble methods could have different results.
- It is designed to dynamically generate high-quality soft targets in a one-stage online knowledge distillation framework.

Uncertainty-aware Score Distribution Learning for Action Quality Assessment

- CVPR 2020, Tang *et al.*

김보은 2020-30224

Perception and Intelligence Laboratory

Action Quality Assessment (AQA)

- Aiming to evaluate how well a specific action is performed



AQA-7 dataset



Suturing

Knot-tying

Needle-passing

JIGSAWS dataset

Why I select this paper?

- Probability model
- Uncertainty (KL-divergence)
- Parametric density estimation

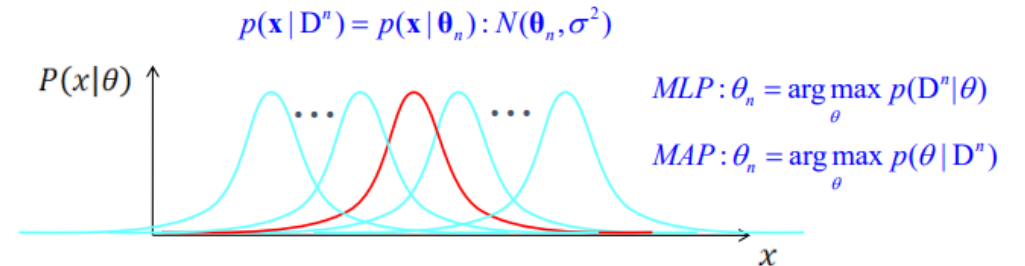
- Relative entropy (or Kullback – Leibler divergence)

$$D_{p\|q} = \sum_{x \in X} p_X(x) \log \left(\frac{p_X(x)}{q_X(x)} \right)$$

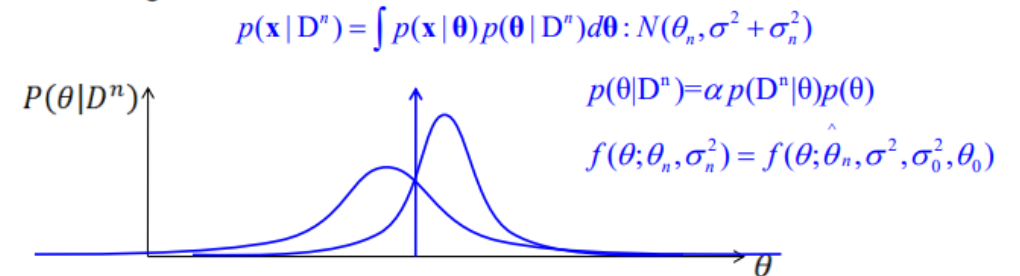
where $p_X(x)$ is probability mass ftn.(pmf), $q_X(x)$ is reference pmf

Maximal Likelihood vs. Bayesian

- MLP, MAP

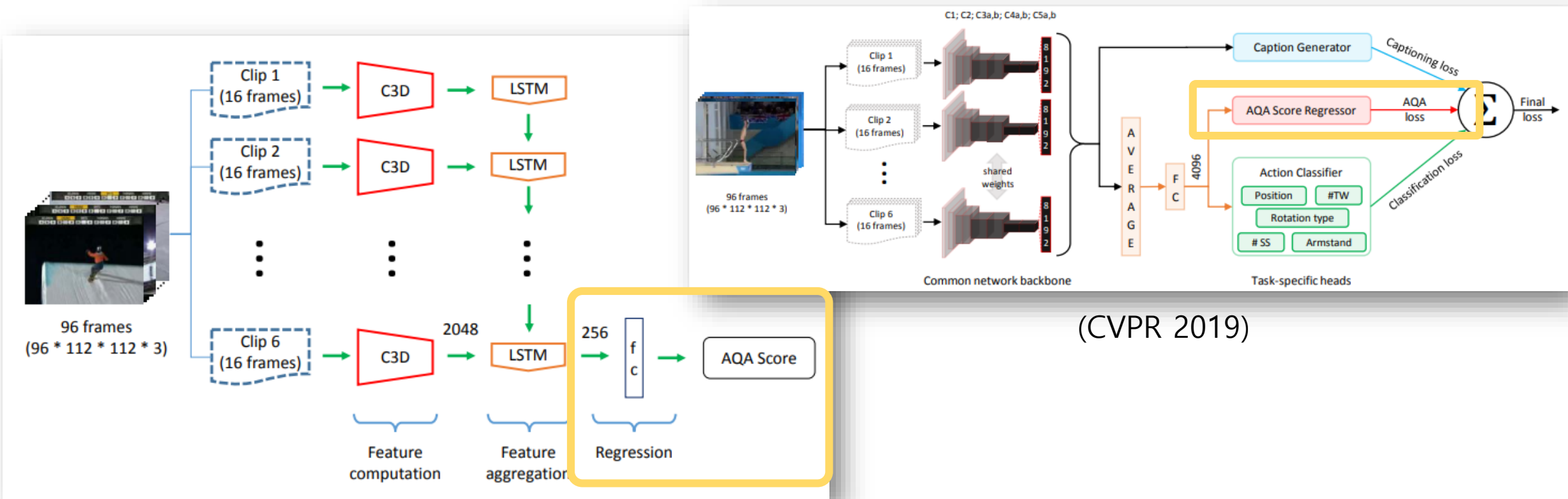


- Bayesian Learning



Previous works

- Regression algorithms which **directly predict evaluation score**
- Ignore the **intrinsic ambiguity** in the score labels caused by multiple judges or their subjective appraisals

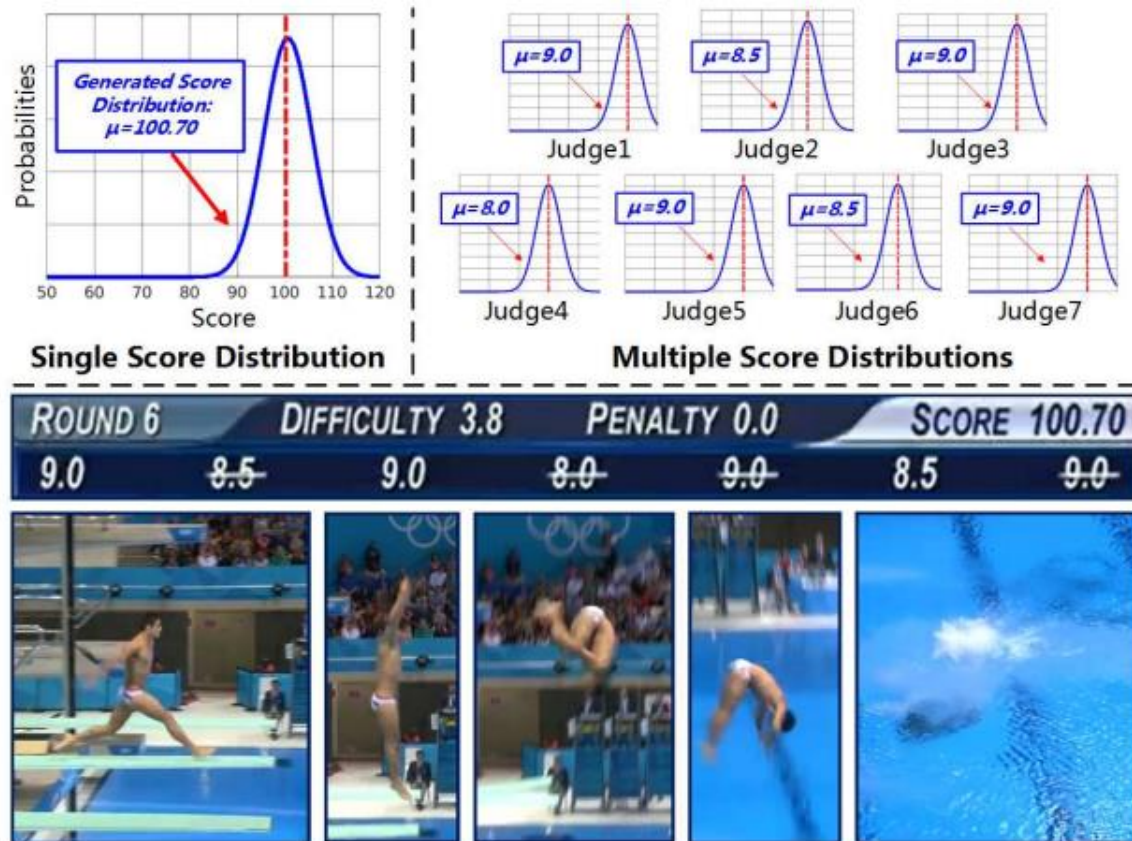


(WACV 2019)

(CVPR 2019)

Main idea

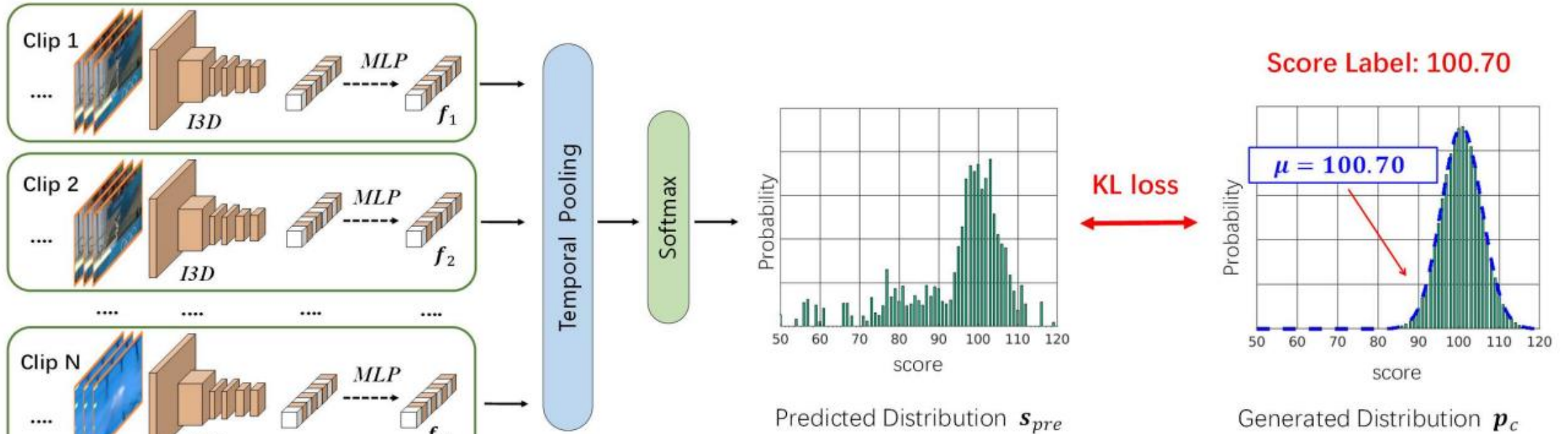
- “Uncertainty-aware score distribution learning”
- Utilize a distribution of different scores as the supervisory signal rather than a single score



$$100.70 = (9.0+9.0+8.5) \times 3.8$$

Assumes the inherent uncertainty of the final score caused by different judges.

Uncertainty-aware score distribution learning (USDL)



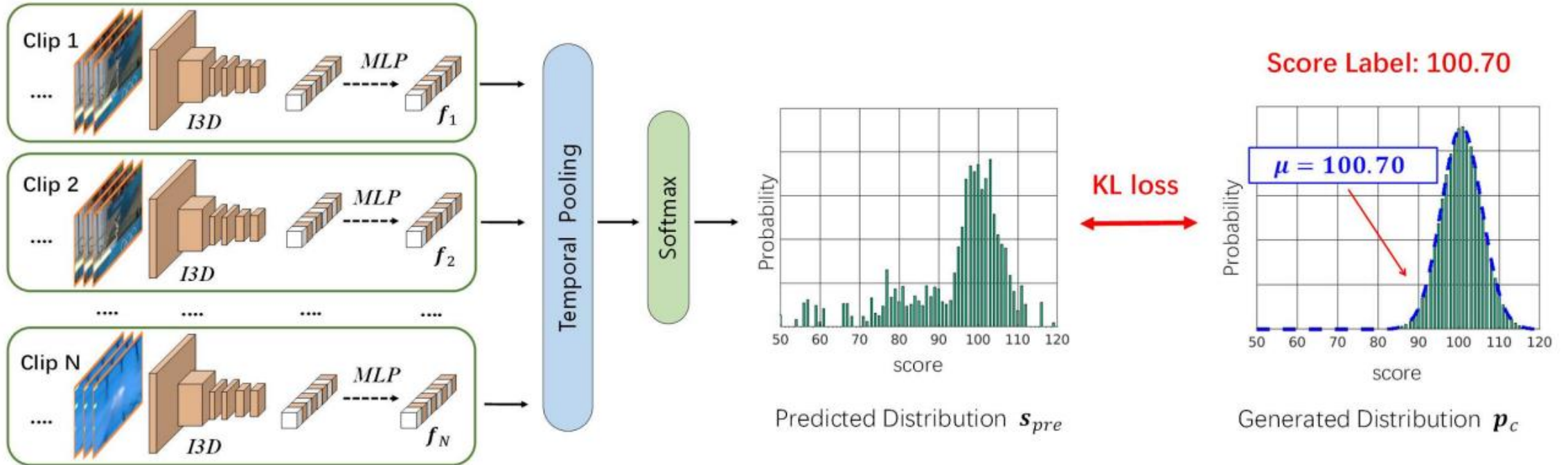
$$g(c) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(c-s)^2}{2\sigma^2}\right)$$

c : score

s : mean = labeled score

σ : hyper parameter

Uncertainty-aware score distribution learning (USDL)

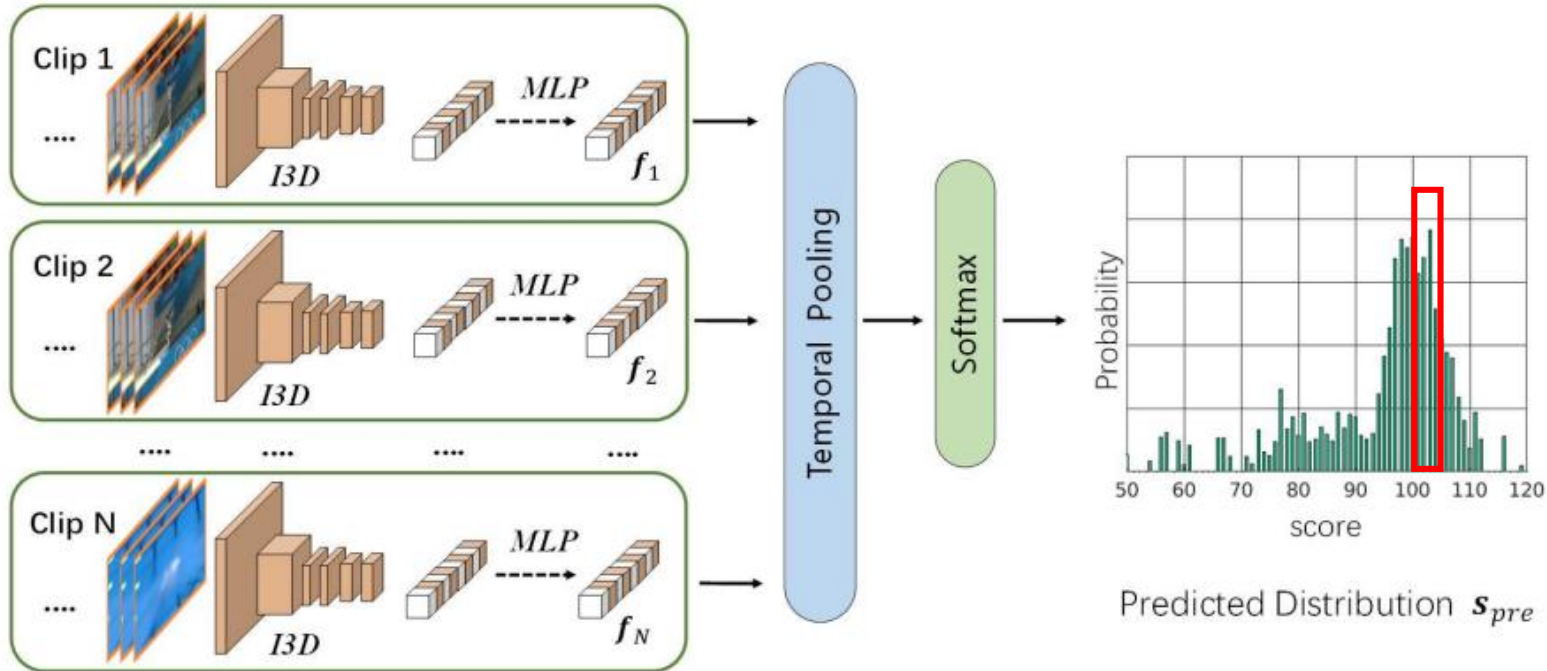


$$\mathbf{s}_{pre} = [s_{pre}(c_1), s_{pre}(c_2), \dots, s_{pre}(c_m)]$$

- Training loss : Kullback-Leibler (KL) divergence between \mathbf{s}_{pre} and \mathbf{P}_c

$$KL\{\mathbf{p}_c || \mathbf{s}_{pre}\} = \sum_{i=1}^m p(c_i) \log \frac{p(c_i)}{s_{pre}(c_i)}$$

Uncertainty-aware score distribution learning (USDL)



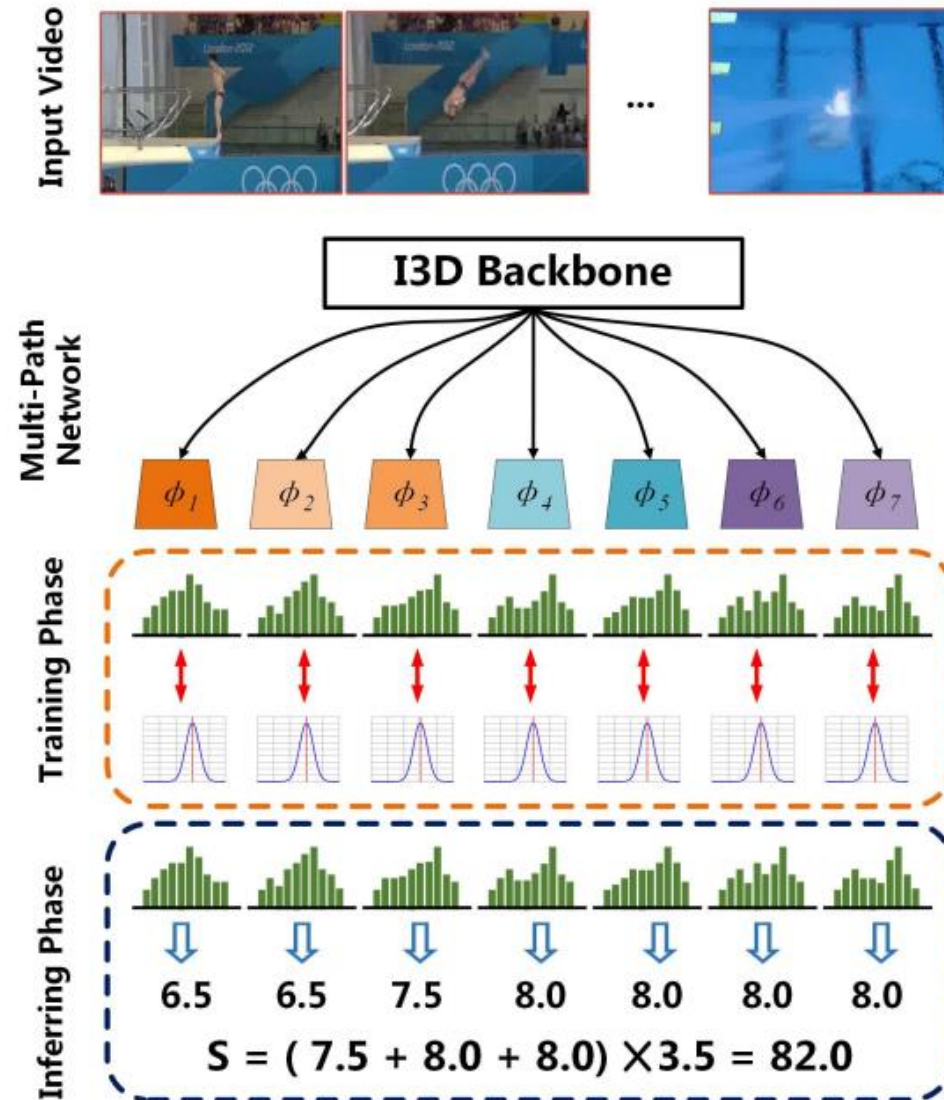
$$\mathbf{s}_{pre} = [s_{pre}(c_1), s_{pre}(c_2), \dots, s_{pre}(c_m)]$$

■ Inferring from score distribution

$$s_{final} = \arg \max_{c_i} \{s_{pre}(c_1), s_{pre}(c_2), \dots, s_{pre}(c_m)\}$$

Multi-path uncertainty-aware score distribution learning (MUSDL)

- Train sub-networks representing judges of different rigor



$$S_{final} = DD \times \sum_{k' \in U} S_{final, k'}^{judge}$$

Results

- Spearman’s rank correlation : (-1, 1), the higher the better

Table 1. Comparisons of action quality assessment accuracy on the AQA-7 dataset.

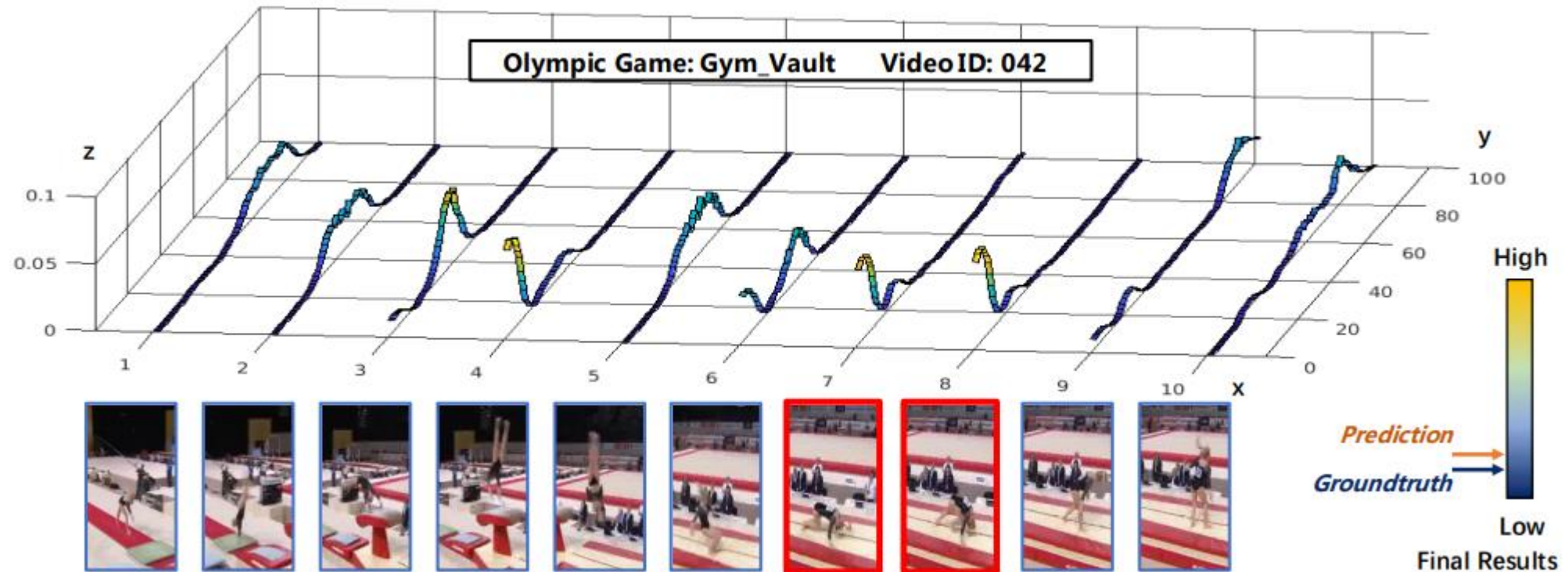
	Diving	Gym Vault	Skiing	Snowboard	Sync. 3m	Sync. 10m	Avg. Corr.
Pose+DCT [22]	0.5300	–	–	–	–	–	–
ST-GCN [38]	0.3286	0.5770	0.1681	0.1234	0.6600	0.6483	0.4433
C3D-LSTM [18]	0.6047	0.5636	0.4593	0.5029	0.7912	0.6927	0.6165
C3D-SVR [18]	0.7902	0.6824	0.5209	0.4006	0.5937	0.9120	0.6937
JRG [16]	0.7630	0.7358	0.6006	0.5405	0.9013	0.9254	0.7849
Ours-Regression	0.7438	0.7342	0.5190	0.5103	0.8915	0.8703	0.7472
Ours-USDL	0.8099	0.7570	0.6538	0.7109	0.9166	0.8878	0.8102

Table 3. Comparisons of performance with existing methods on the MTL-AQA dataset.

Method	Sp. Corr.
Pose+DCT [22]	0.2682
C3D-SVR [18]	0.7716
C3D-LSTM [18]	0.8489
MSCADC-STL [19]	0.8472
C3D-AVG-STL [19]	0.8960
MSCADC-MTL [19]	0.8612
C3D-AVG-MTL [19]	0.9044
Ours-Regression	0.8905
Ours-MUSDL	0.9273

Results

- Spearman's rank correlation : (-1, 1), the higher the better



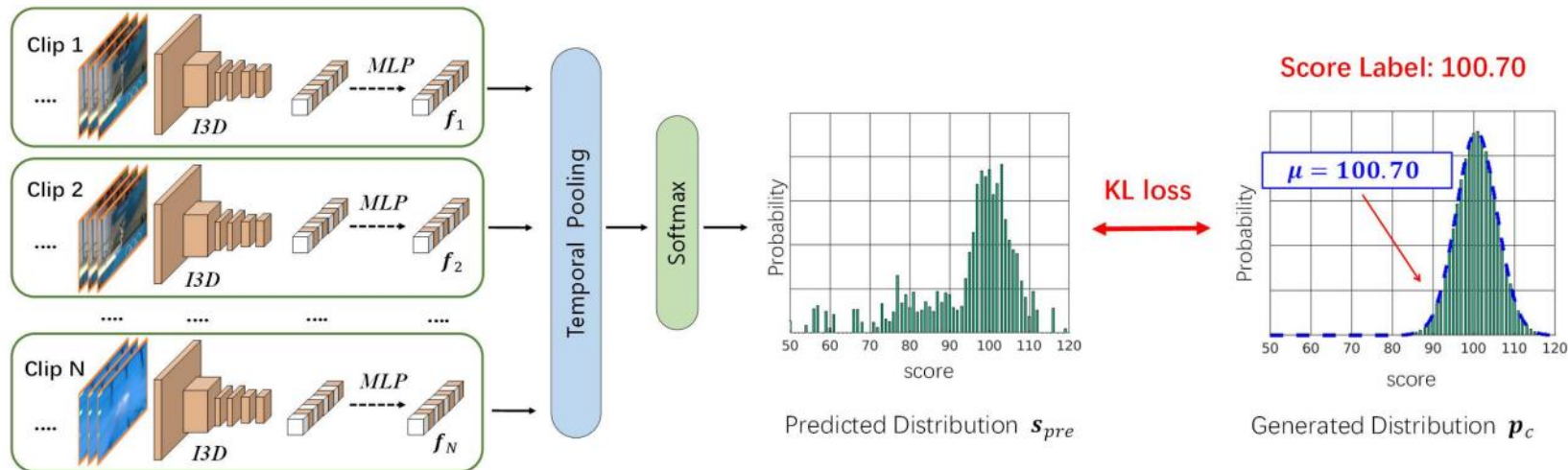
fell to the ground
: low score prediction

Contribution

“Uncertainty-aware score distribution learning” rather than “directly predict evaluation score”

→ Learn inherent uncertainty of the score caused by different judges

→ Set new state-of-the-art performance



Thank you.

Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech

2021-27680 김세민



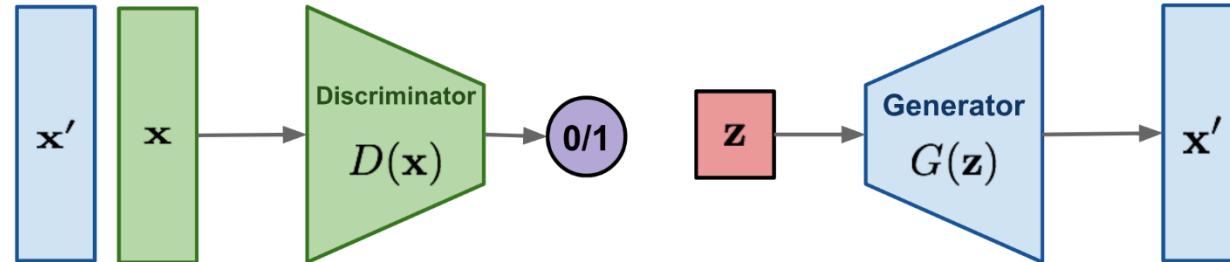
VITS

- Variational Inference with adversarial learning for end-to-end Text-to-Speech
- 기존의 mel-spectrogram -> audio 의 two-stage TTS만큼 natural 하면서 parallel 하게 sampling이 가능한 TTS 모델이다.
- 그 과정에서 Flow, GAN, Conditional VAE 등의 generative model을 복합적으로 이용한다.

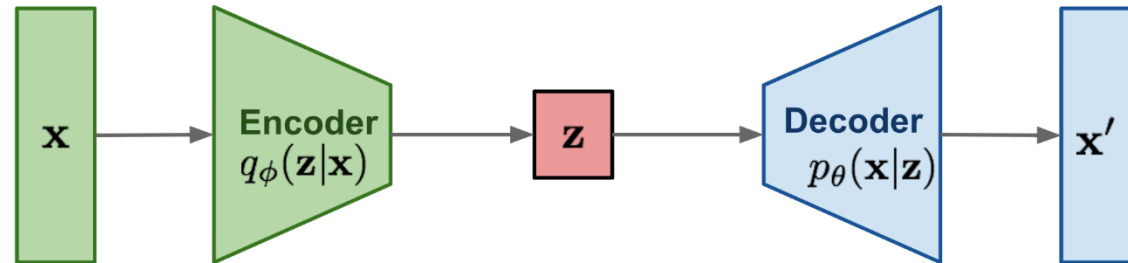


Generative Models

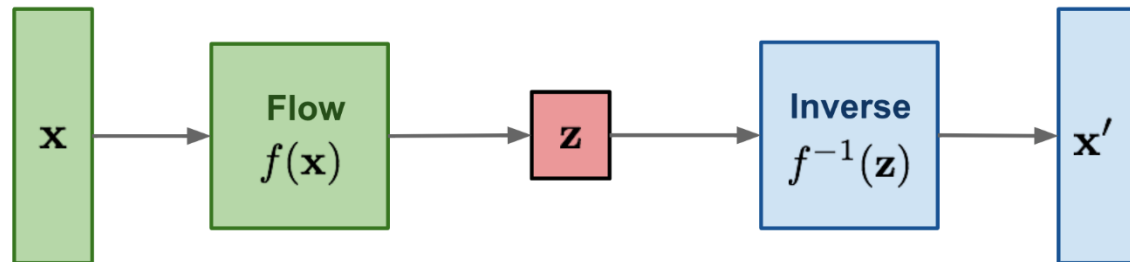
GAN: minimax the classification error loss.



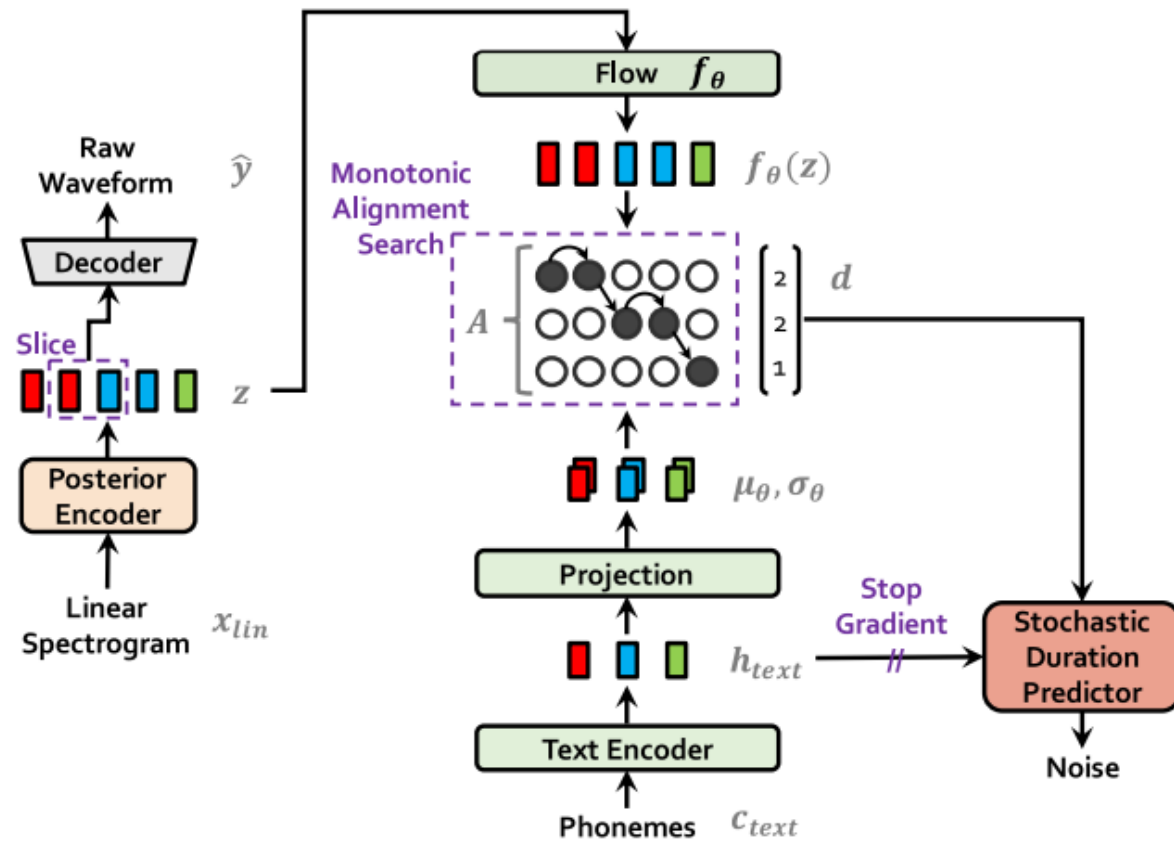
VAE: maximize ELBO.



Flow-based generative models: minimize the negative log-likelihood



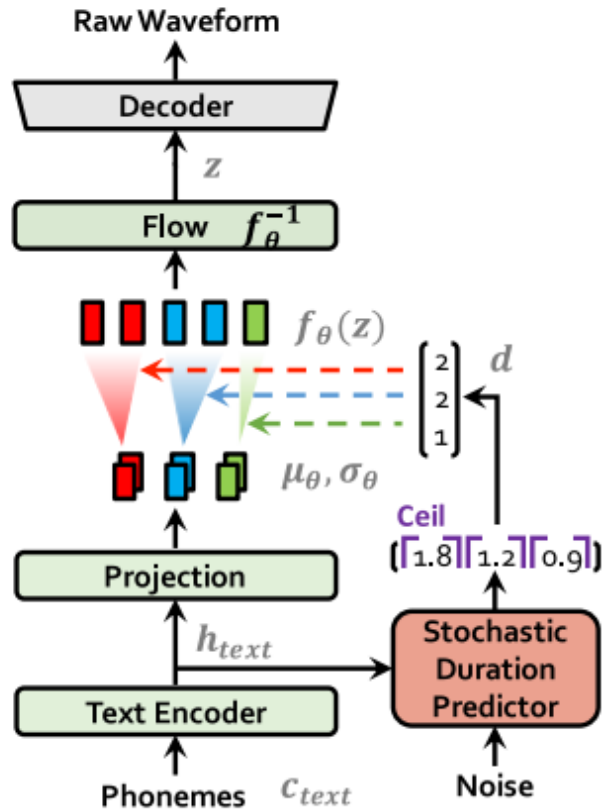
Model(Training)



X: spectrogram
Z: latent variable
Y: Ground Truth audio
Y-hat: generated audio
C: phoneme of text



Model(Inference)



Text를 input으로 넣으면 Encoder를 통해 parameterized 되고 이를 duration predictor의 결과에 따라 길이를 늘려준 다음, f인버스의 결과로 Decoder를 통해 다시 audio로 inference함.



Result

LJ Speech Dataset

Model	MOS (CI)
Ground Truth	4.46 (± 0.06)
Tacotron 2 + HiFi-GAN	3.77 (± 0.08)
Tacotron 2 + HiFi-GAN (Fine-tuned)	4.25 (± 0.07)
Glow-TTS + HiFi-GAN	4.14 (± 0.07)
Glow-TTS + HiFi-GAN (Fine-tuned)	4.32 (± 0.07)
VITS (DDP)	4.39 (± 0.06)
VITS	4.43 (± 0.06)

VCTK Dataset(Multi Speaker)

Model	MOS (CI)
Ground Truth	4.38 (± 0.07)
Tacotron 2 + HiFi-GAN	3.14 (± 0.09)
Tacotron 2 + HiFi-GAN (Fine-tuned)	3.19 (± 0.09)
Glow-TTS + HiFi-GAN	3.76 (± 0.07)
Glow-TTS + HiFi-GAN (Fine-tuned)	3.82 (± 0.07)
VITS	4.38 (± 0.06)

Synthesis Speed

Model	Speed (kHz)	Real-time
Glow-TTS + HiFi-GAN	606.05	$\times 27.48$
VITS	1480.15	$\times 67.12$
VITS (DDP)	2005.03	$\times 90.93$



Top-tier Conference Paper Presentation

“Normalization Matters in Weakly Supervised Object Localization”

(ICCV 2021, Jeesoo Kim et al.)

Department of Applied Bioengineering,

Graduate School of Convergence Science and Technology,

Seoul National University

Sihwan Kim

(imksh0707@snu.ac.kr)



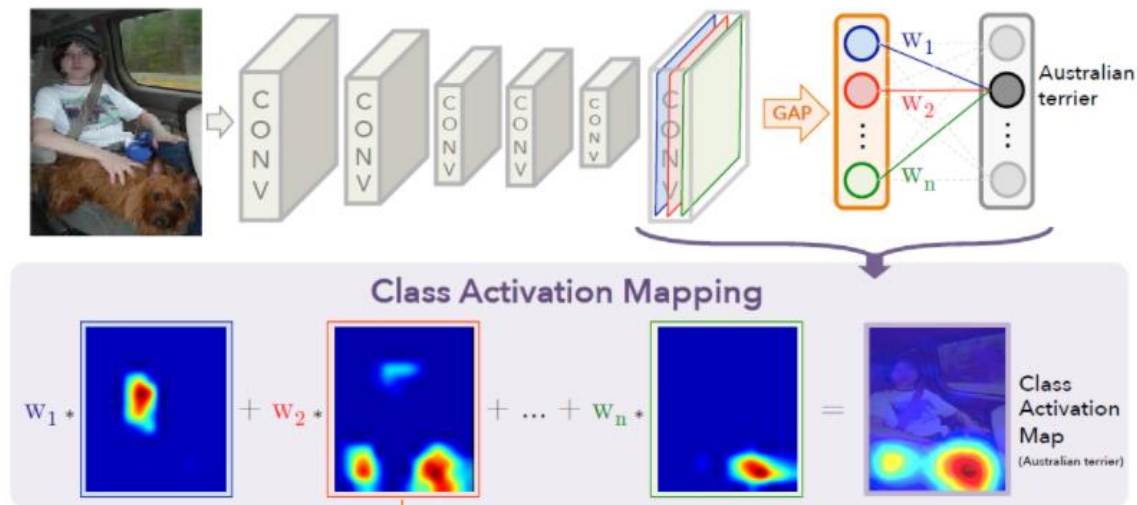
Contents

- Purposes
- Background Knowledge
- Results & Discussion
 - Qualitative Analysis
 - Quantitative Analysis
- Conclusions



Purpose & Background Knowledge

- Purpose
 - To suggest the Optimal Normalization Methods for CAM-based Localization Method.
- What's the Class Activation Map (CAM)?



$$\mathbf{F}_c = \frac{1}{K} \sum_{i=1}^K \mathbf{w}_i^c \cdot f_i(X) \quad \rightarrow \quad \mathbf{F}'_c = H(\mathbf{F}_c)$$

K : Num. of Channels

H : Normalization Function

F' : Score Map

C : class Index

- What kinds of CAM-based *WSOL Methods Exist?

- a) **Class Activation Mapping (CAM)**

- The first approach using the activated convolutional feature as a score map to locate an object in an image.

- b) **Hide-and-Seek (HaS)**

- Makes a grid in an image and randomly erases multiple patches. The model struggles to make a correct decision with the corrupted image and this induce the feature of the model to be activated in the location of the target object.

- c) **Attention-based Dropout Layer (ADL)**

- Uses two branches which adversarially get rid of the highlighted activated region from each other. This approach is different from HaS in that the feature is erased instead of the image itself.

- d) **CutMix**

- Patches from training images are cut and pasted to one another and this helps the model to capture less discriminative parts of the target object.



Materials & Methods

Types of Normalization in the Paper

a) Min-max

$$F' = \frac{F - \min(F)}{\max(F) - \min(F)}$$

b) PaS

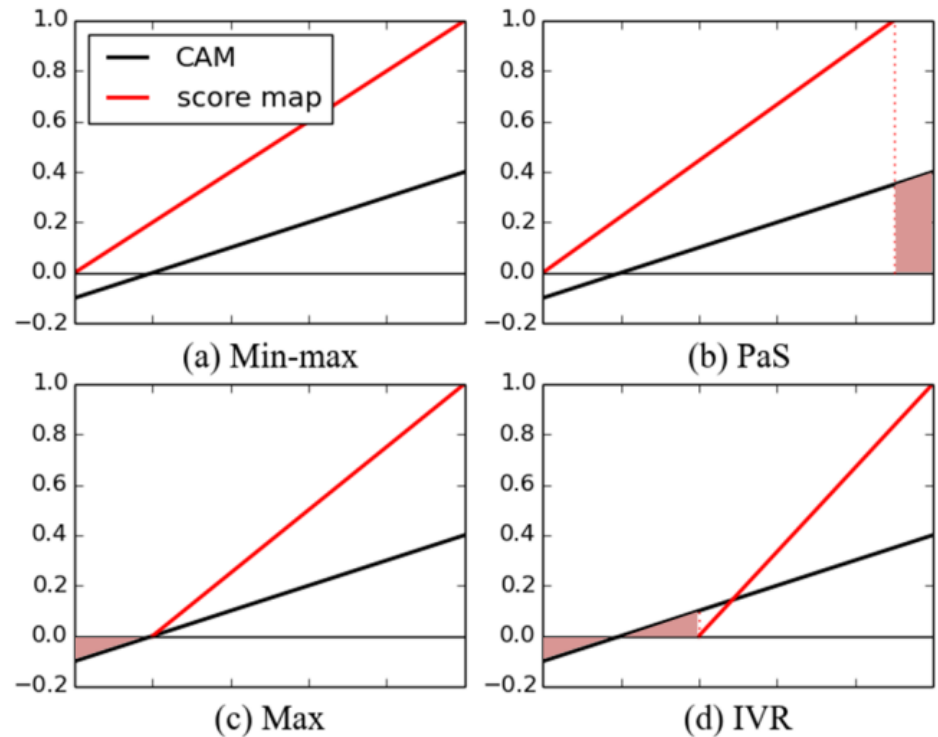
$$F' = \frac{F - \min(F)}{\text{Pct}_p(F - \min(F))}$$

c) Max

$$F' = \frac{F}{\max(F)}$$

d) IVR

$$F' = \frac{F - \text{Pct}_p(F)}{\max(F - \text{Pct}_p(F))}$$



➤ Assume that the region colored in red is excluded from the final score map.

Results & Discussion: Qualitative Analysis

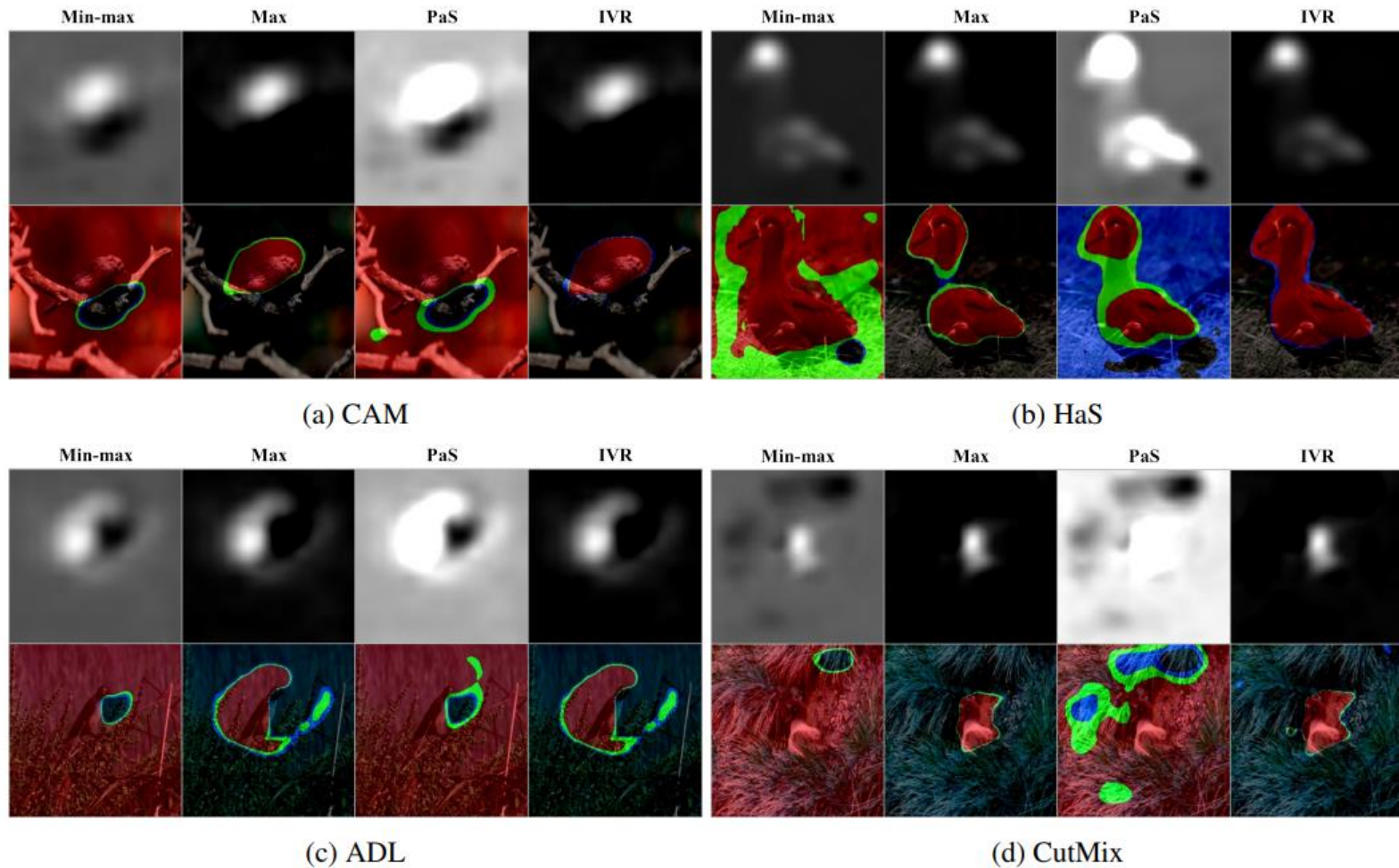


Figure: Visualization of score maps and activated areas by the optimal thresholds in CAM, HaS, ADL, and CutMix on CUB dataset. Red, green and blue area are regions extracted by the optimal thresholds of IoU 0.3, 0.5 and 0.7.

Results & Discussion: Quantitative Analysis

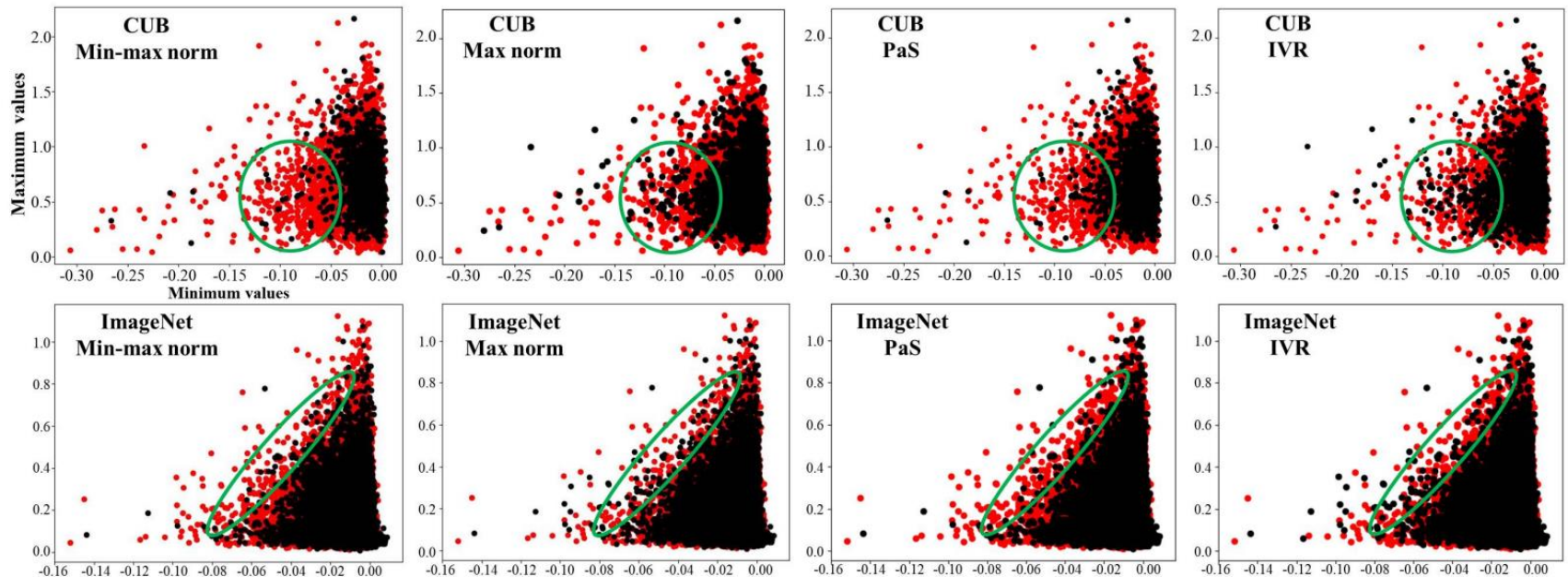
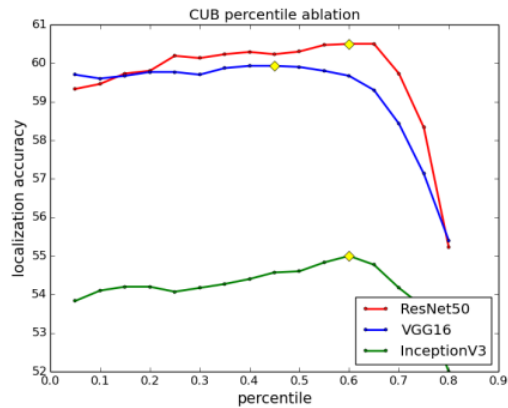
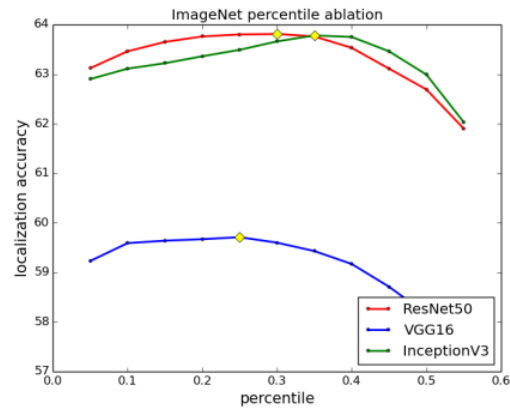


Figure: Distribution of minimum and maximum values from class activation map F of all test images. Each point represents a single image whose values from horizontal and vertical axes are minimum value and maximum value respectively. Black dots correspond to correctly localized images while red dots are not. Note that IVR shows overall higher density of positive samples (black dots) than other methods as highlighted with green circles.

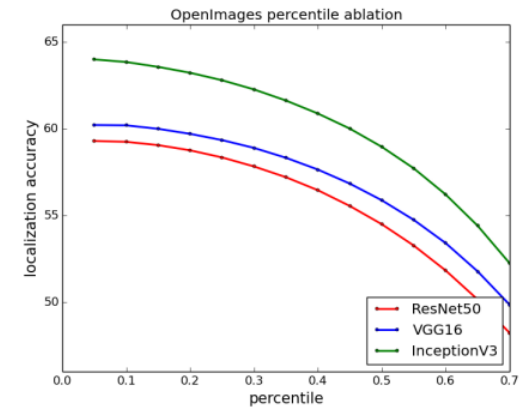
Results & Discussion



(a) CUB



(b) ImageNet



(c) OpenImages

Figure: Localization accuracy measured with different percentile values when using IVR. The evaluation has been done only with CAM in the validation set of every datasets. To keep simplicity, the best percentile value in each architecture and dataset has been used in all other WSOL methods.

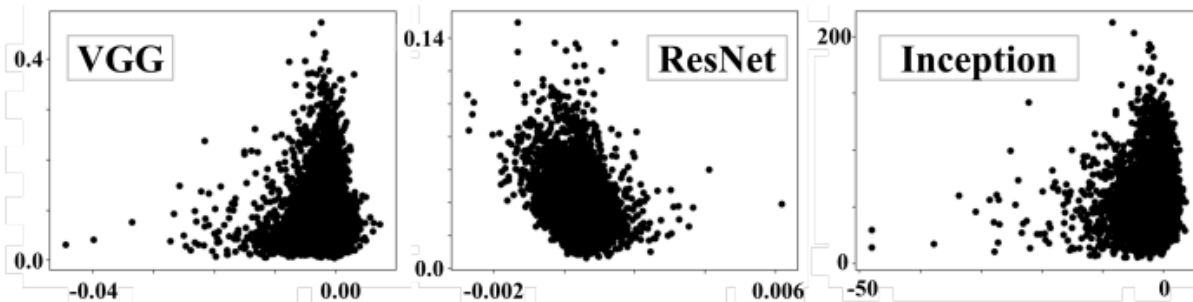


Figure: Distribution of minimum and maximum values from class activation map F of all test images in OpenImages. Compared to CUB and ImageNet, the distribution of maximum values is more influential.

Conclusions

- ✓ New and effective normalization method along with a solid evaluation with many other possible normalization methods are proposed.
- ✓ Point out that the normalization method should be selected according to the traits of the dataset
- ✓ For future works in WSOL, we suggest that even though many WSOL methods successfully improved the performance in a dataset like CUB, a new perspective which will also work in real world datasets such as ImageNet and OpenImages is still in need.



Efficient and Scalable Bayesian Neural Nets with Rank-1 Factors

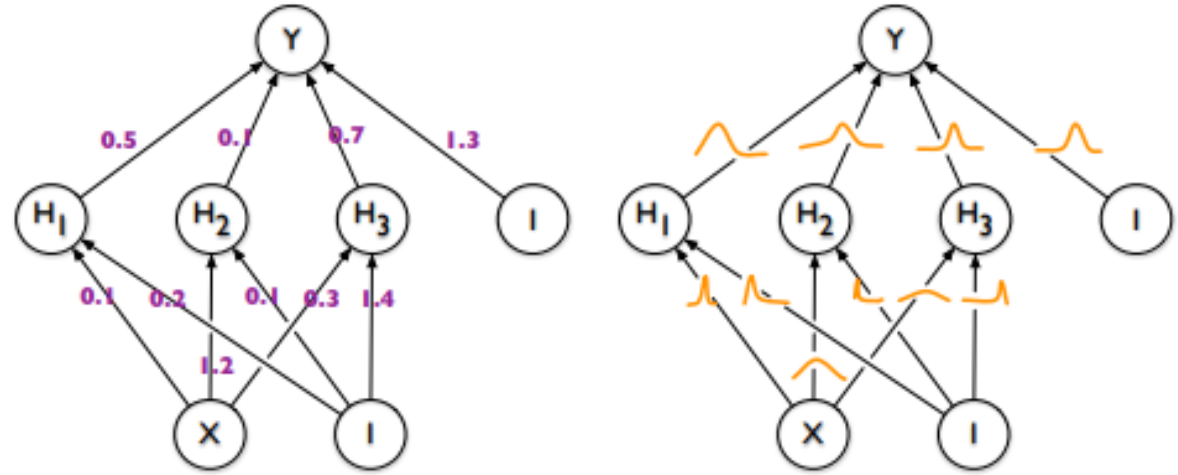
ICML 2020

Google Brain, Duke university

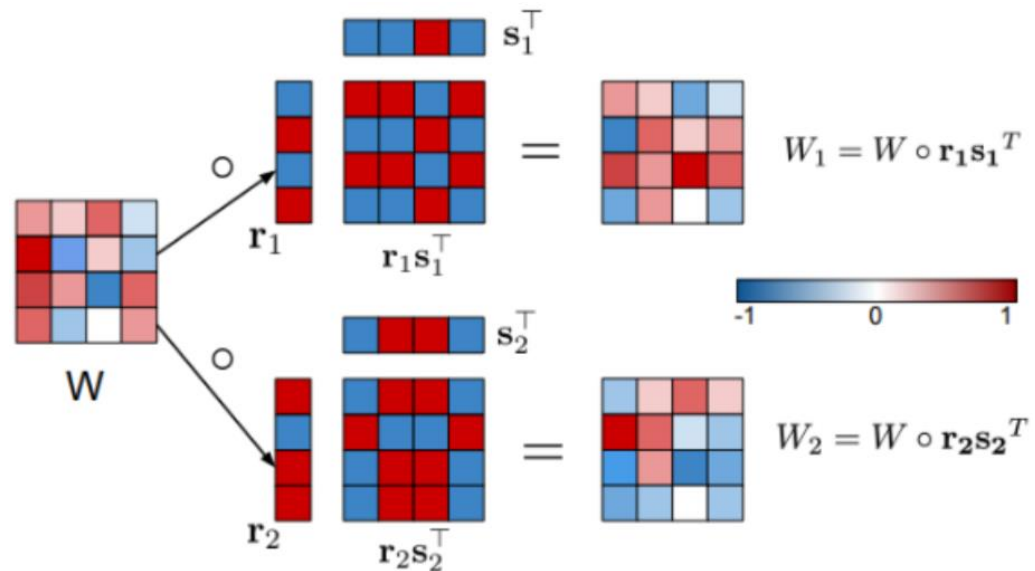
Michael W. Dusenberry^{*†1} **Ghassen Jerfel**^{*12} **Yeming Wen**¹³ **Yi-An Ma**¹⁴ **Jasper Snoek**¹
Katherine Heller¹² **Balaji Lakshminarayanan**¹ **Dustin Tran**¹

Main keyword

- Bayesian Neural Network



- BatchEnsemble



Bayesian Neural Network

- Neural network for uncertainty



Dog



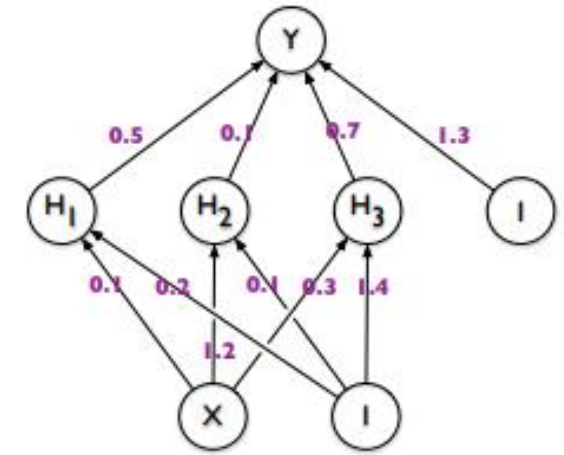
Cat



???

Bayesian Neural Network

- Neural network for uncertainty



Traditional neural network



→ Dog



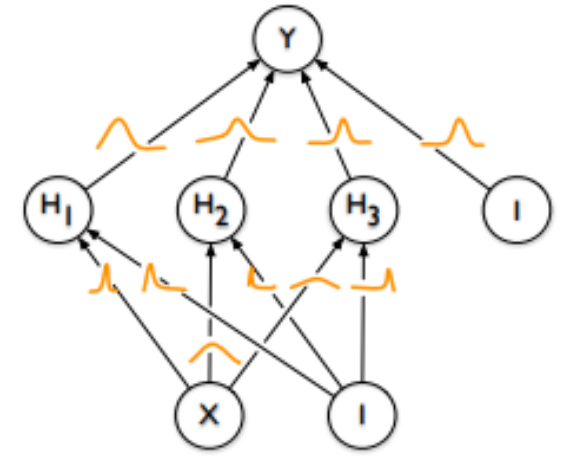
→ Cat



→ Dog

Bayesian Neural Network

- Neural network for uncertainty



Bayesian neural network



→ Dog



→ Cat

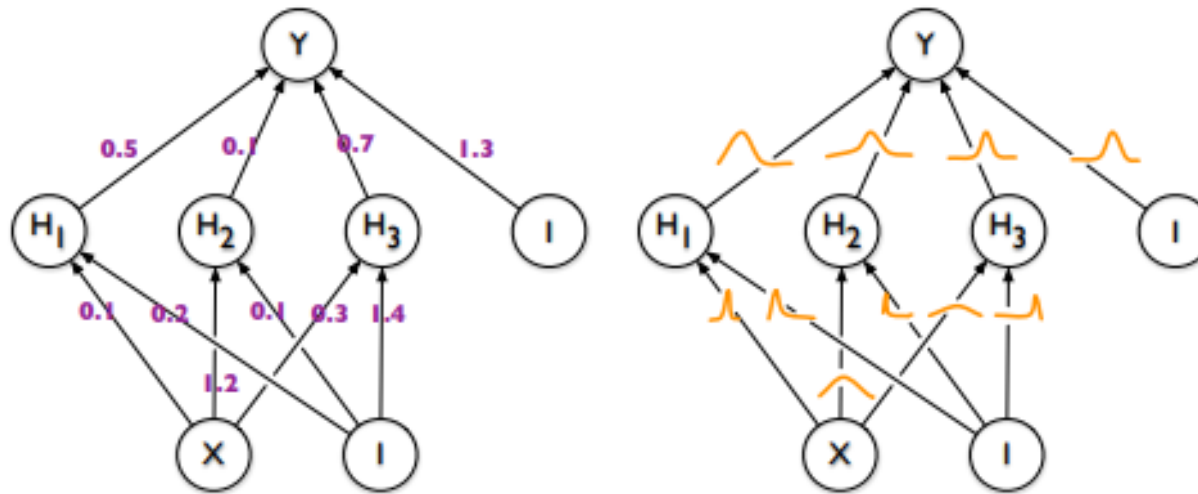


→ Dog
with 55%

Bayesian Neural Network

- Weight extracted from distribution (e.g. gaussian)
- Learning mean and std rather than weight

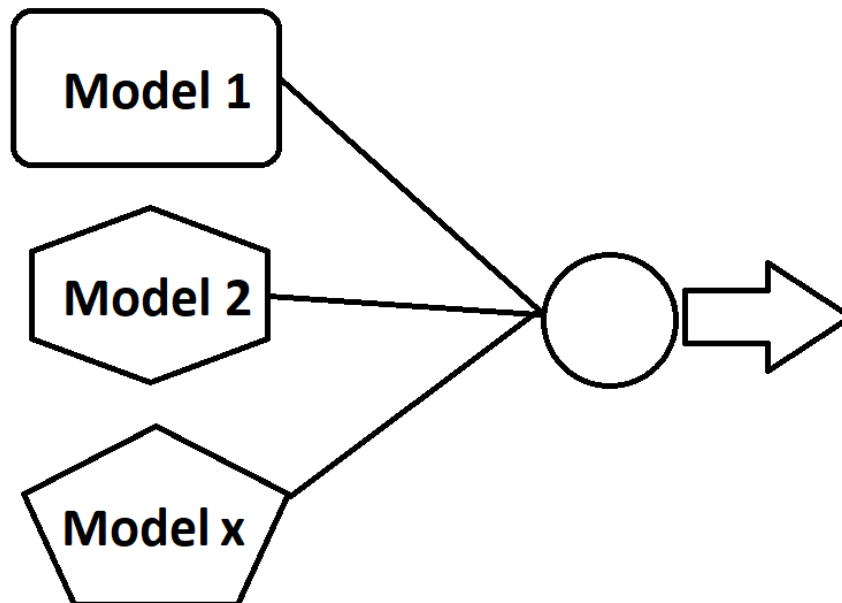
$$X \sim N(\mu, \sigma^2)$$



$$\mathcal{L} = -\frac{N}{B} \sum_{b=1}^B \mathbb{E}_{q(\mathbf{W})} [\log p(y_b | \mathbf{x}_b, \mathbf{W})] + \text{KL}(q(\mathbf{W}) || p(\mathbf{W}))$$

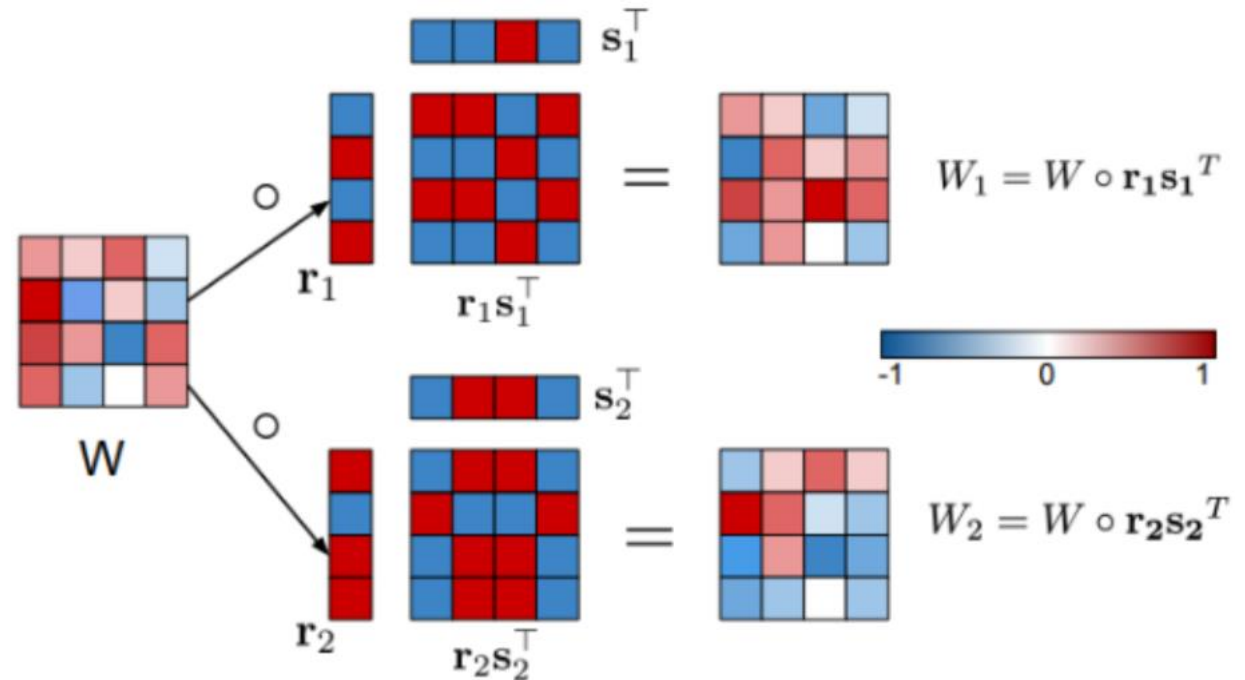
BatchEnsemble

- Ensemble method
 - Getting better performance by combining several models
 - **Weakness: Learning parameters highly increase**



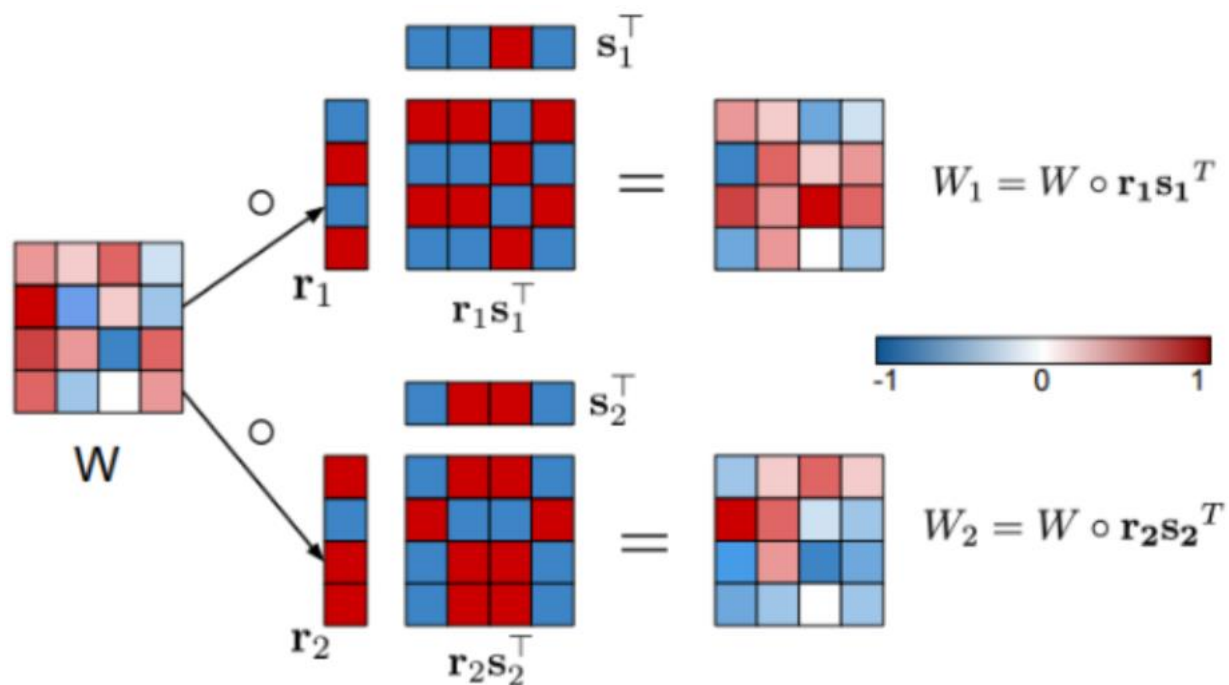
BatchEnsemble

- Shared weight parameter W
- Additional parameter vector r and s
- Ensemble W_1 and $W_2 \dots$



Proposed method

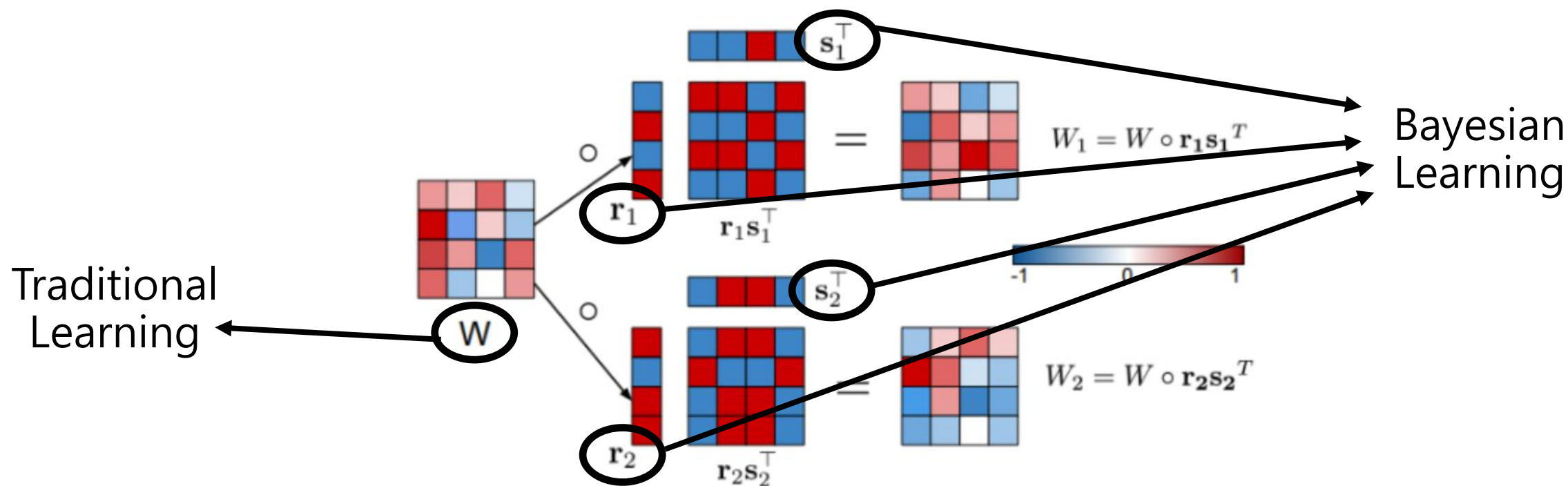
- BatchEnsemble on Bayesian Neural Network



$$\mathcal{L} = -\frac{N}{B} \sum_{b=1}^B \mathbb{E}_{q(r)q(s)} [\log p(y_b | x_b, W, r, s)] + KL(q(r) || p(r)) + KL(q(s) || p(s)) - \log p(w)$$

Proposed method

- BatchEnsemble on Bayesian Neural Network



$$\mathcal{L} = -\frac{N}{B} \sum_{b=1}^B \mathbb{E}_{q(r)q(s)} [\log p(y_b | x_b, W, r, s)] + KL(q(r) || p(r)) + KL(q(s) || p(s)) - \log p(w)$$

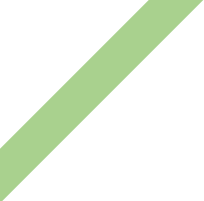
Result

- # parameter is not increase with high performance on BNN

Efficient and Scalable Bayesian Neural Nets with Rank-1 Factors

Method		NLL(↓)	Accuracy(↑)	ECE(↓)	cNLL / cA / cECE	mCE(↓)	# Parameters
Deterministic		0.943	76.1	0.0392	3.20 / 40.5 / 0.105	75.34	25.6M
BatchEnsemble		0.951	76.5	0.0532	3.23 / 41.4 / 0.120	74.14	25.8M
Rank-1 BNN	Gaussian	0.886	77.3	0.0166	2.95 / 42.9 / 0.054	72.12	26.0M
	Cauchy ^(4 samples)	0.897	77.2	0.0192	2.98 / 42.5 / 0.059	72.66	26.0M
Deep Ensembles	ResNet-50	0.877	77.5	0.0305	2.98 / 42.1 / 0.050	73.25	146.7M
MCMC BNN ¹	9 MC samples	0.888	77.1	-	-	-	230.4
MCMC BNN ²	30 MC samples	0.883	77.5	-	-	-	768M

THANKS



Bias-Variance Reduced Local SGD for Less Heterogeneous Federated Learning (ICML2021)

지능정보융합학과 2021-29046 김준엽



Reason for selection

- 수업에서 배운 optimization과 관련된 논문을 ICML2021에서 search
- 배운 optimization은 대부분 convex 함수를 위한 기법
- 관심 Domain인 Big-data 분야에는 고차원, non-convex 성질을 가지는 경우가 많은데,이런 데이터에 더 적합한 sgd를 찾아보고 싶었음
- Minibatch Stochastic Gradient Descent(SGD) (Dekel et al., 2012), Minibatch Stochastic Recursive Gradient algorithm (SARAH) (Nguyen et al., 2017; 2019), Local SGD Converges Fast and Communicates Little(Sebastian U. Stich, 2019) 등 계속해서 더 좋은 sgd 개발

Introduction

- 최적화 문제는 시간이 오래걸림 -> 분산 학습, 병렬 컴퓨팅
- Minibatch SGD는 통신비용이 높고(각 작업자가 미니배치 기울기 계산 후 평균으로 글로벌 모델을 업데이트), 통신 병목 현상에 취약함
- Local SGD(Parallel Restart SGD)는 통신 병목 현상을 극복하기 위한 방법(각 작업자가 자신의 로컬 데이터셋을 기반으로 로컬 모델을 독립적으로 업데이트하고, 주기적으로 모델을 통신한 뒤 평균하는 방식)
- 일반적으로 local SGD의 통신 복잡성과 정확성은 non-local SGD(minibatch)보다 나쁨.
- Woodworth et al. Is local sgd better than minibatch sgd?(2020)에서 heterogeneity가 작을 때 local SGD가 non-local SGD보다 성능이 뛰어남(더 낮은 통신복잡성)을 증명

Problem definition

We want to minimize nonconvex smooth objective

$$f(x) := \frac{1}{P} \sum_{p=1}^P f_p(x), \text{ where } f_p(x) := \mathbb{E}_{z \sim D_p}[\ell(x, z)]$$

- 논문의 목표 : 보다 완화된 조건에서 non-local 및 기존 local algorithm을 능가하는 local algorithm을 찾자.
- 각 작업자 p 는 통신이 없으면, 자신의 데이터 분포 D_p 에만 액세스할 수 있다고 가정.

Assumption 1 (Heterogeneity). $\{f_p\}_{p=1}^P$ is second-order ζ -heterogeneous, i.e., for any $p, p' \in [P]$,

$$\|\nabla^2 f_p(x) - \nabla^2 f_{p'}(x)\| \leq \zeta, \forall x \in \mathbb{R}^d.$$

Assumption 2 (Smoothness). For any $p \in [P]$ and $z \in \text{supp}(D_p)$, $\ell(\cdot, z)$ is L -smooth, i.e.,

$$\|\nabla \ell(x, z) - \nabla \ell(y, z)\| \leq L\|x - y\|, \forall x, y \in \mathbb{R}^d.$$

Assumption 3 (Existence of global optimum). f has a global minimizer $x_* \in \mathbb{R}^d$.

Assumption 4 (Bounded gradient variance). For every $p \in [P]$,

$$\mathbb{E}_{z \sim D_p} \|\nabla \ell(x, z) - \nabla f_p(x)\|^2 \leq \sigma^2.$$

Assumption 4 says that the variance of stochastic gradient is bounded for every local objective.

Building blocks of Bias-Variance Reduced Local SGD

1) Localization

- Regularization 기법중 하나. 논문에서는 통신 비용 줄이기 위해 고려.

2) Bias reduction

- local gradients의 bias를 줄인다
- 주기적으로 $\nabla f(x)$ 를 계산해서 estimator의 bias을 최소화하고, 이로써 second-order heterogeneity ζ 가 작지 않은 경우에도 vanilla non-local 및 local GD보다 빠르게 수렴할 수 있게 함

Building blocks of Bias-Variance Reduced Local SGD

3) Stochastization(확률화)

- 결정론적 방법은 대규모 최적화에서 단일 업데이트에 막대한 계산 비용이 필요
- GD to SGD (계산 비용 감소)

4) Variance reduction

- Convex뿐만 아니라 Non-convex function에도 사용가능한 기법으로, Stochastization로 발생하는 분산을 줄이고 주기적으로 계산되는 local gradients를 사용하여 계산 효율성을 향상

Bias-Variance Reduced Local SGD Algorithm

Algorithm 1 Local GD($\tilde{x}_0, \eta, B, b, K, T$)

```

1: for  $t = 1$  to  $T$  do
2:   for  $p = 1$  to  $P$  in parallel do
3:     Set  $x_0^{(p)} = \tilde{x}_{t-1}$ .
4:     for  $k = 1$  to  $K$  do
5:       Update  $x_k^{(p)} = x_{k-1}^{(p)} - \eta \nabla f_p(x_{k-1}^{(p)})$ 
6:     end for
7:   end for
8:   Communicate  $\{x_t^{(p)}\}_{p=1}^P$ .
9:    $\tilde{x}_t = \frac{1}{P} \sum_{p=1}^P x_{\hat{k}}^{(p)}$  ( $\hat{k} \sim \text{Unif}[K]$ ).
10: end for
11: Return:  $\tilde{x}_{\hat{t}}$  ( $\hat{t} \sim \text{Unif}[T]$ ).

```

Algorithm 2 BVR-L-SGD($\tilde{x}_0, \eta, b, \tilde{b}, K, T, S$)

```

1: for  $s = 1$  to  $S$  do
2:   for  $p = 1$  to  $P$  in parallel do
3:     if  $\tilde{b} \geq \frac{1}{P} \sum_{p=1}^P \#\text{supp}(D_p)$  then
4:        $\tilde{\nabla}^{(p)} = \nabla f_p(\tilde{x}_{s-1})$ .
5:     else
6:        $\tilde{\nabla}^{(p)} = \frac{1}{b} \sum_{l=1}^{\tilde{b}} \nabla \ell(\tilde{x}_{s-1}, z_l)$  ( $z_l \stackrel{i.i.d.}{\sim} D_p$ ).
7:     end if
8:   end for
9:   Communicate  $\{\tilde{\nabla}^{(p)}\}_{p=1}^P$ , set  $\tilde{v}_0 = \frac{1}{P} \sum_{p=1}^P \tilde{\nabla}^{(p)}$ .
10:  Set  $x_0 = x_{-1} = \tilde{x}_{s-1}$ .
11:  for  $t = 1$  to  $T$  do
12:    for  $p = 1$  to  $P$  in parallel do
13:       $g_t^{(p)}(x_{t-1}) = \frac{1}{Kb} \sum_{l=1}^{Kb} \nabla \ell(x_{t-1}, z_l)$ ,
14:       $g_t^{(p)}(x_{t-2}) = \frac{1}{Kb} \sum_{l=1}^{Kb} \nabla \ell(x_{t-2}, z_l)$ 
15:      for  $z_l \stackrel{i.i.d.}{\sim} D_p$ .
16:       $\tilde{v}_t^{(p)} = g_t^{(p)}(x_{t-1}) - g_t^{(p)}(x_{t-2}) + \tilde{v}_{t-1}^{(p)}$ .
17:    end for
18:    Communicate  $\{\tilde{v}_t^{(p)}\}_{p=1}^P$ , set  $\tilde{v}_t = \frac{1}{P} \sum_{p=1}^P \tilde{v}_t^{(p)}$ .
19:    for  $p = 1$  to  $P$  in parallel do
20:       $x_t^{(p)}, x_t^{(p),\text{out}} =$ 
21:        Local-Routine( $p, x_{t-1}, \eta, \tilde{v}_t, b, K$ ).
22:    end for
23:    Communicate  $\{x_t^{(p)}\}_{p=1}^P$  and  $\{x_t^{(p),\text{out}}\}_{p=1}^P$ .
24:    Set  $x_t = x_t^{(\hat{p})}$  and  $x_t^{\text{out}} = x_t^{(\hat{p}),\text{out}}$  ( $\hat{p} \sim \text{Unif}[P]$ ).
25:  end for
26:  Set  $\tilde{x}_s = x_T$  and  $\tilde{x}_s^{\text{out}} = x_{\hat{t}}^{\text{out}}$  ( $\hat{t} \sim \text{Unif}[T]$ ).
27: end for
28: Return:  $\tilde{x}_s^{\text{out}} = \tilde{x}_{\hat{s}}^{\text{out}}$  ( $\hat{s} \sim \text{Unif}[S]$ ).

```

Algorithm 3 Local-Routine(p, x_0, η, v_0, b, K)

```

1: Set  $x_0^{(p)} = x_{-1}^{(p)} = x_0$ .
2: for  $k = 1$  to  $K$  do
3:    $g_k^{(p)}(x_{k-1}^{(p)}) = \frac{1}{b} \sum_{l=1}^b \ell(x_{k-1}^{(p)}, z_l)$ ,  $g_k^{(p)}(x_{k-2}^{(p)}) =$ 
4:      $\frac{1}{b} \sum_{l=1}^b \ell(x_{k-2}^{(p)}, z_l)$  ( $z_l \stackrel{i.i.d.}{\sim} D_p$ ).
5:    $v_k^{(p)} = g_k^{(p)}(x_{k-1}^{(p)}) - g_k^{(p)}(x_{k-2}^{(p)}) + v_{k-1}^{(p)}$ .
6:   Update  $x_k^{(p)} = x_{k-1}^{(p)} - \eta v_k^{(p)}$ 
7: end for
8: Return:  $x_K^{(p)}, x_{\hat{k}}^{(p)}$  ( $\hat{k} \sim \text{Unif}[K]$ ).

```

Results

- 완화된 조건인 heterogeneity가 클 때 기존 local sgd들의 성능은 크게 저하되는 반면, BVR-L-SGD는 그 정도가 훨씬 작음.
- BVR-L-SGD는 기존의 non-local, local sgd의 성능을 능가.

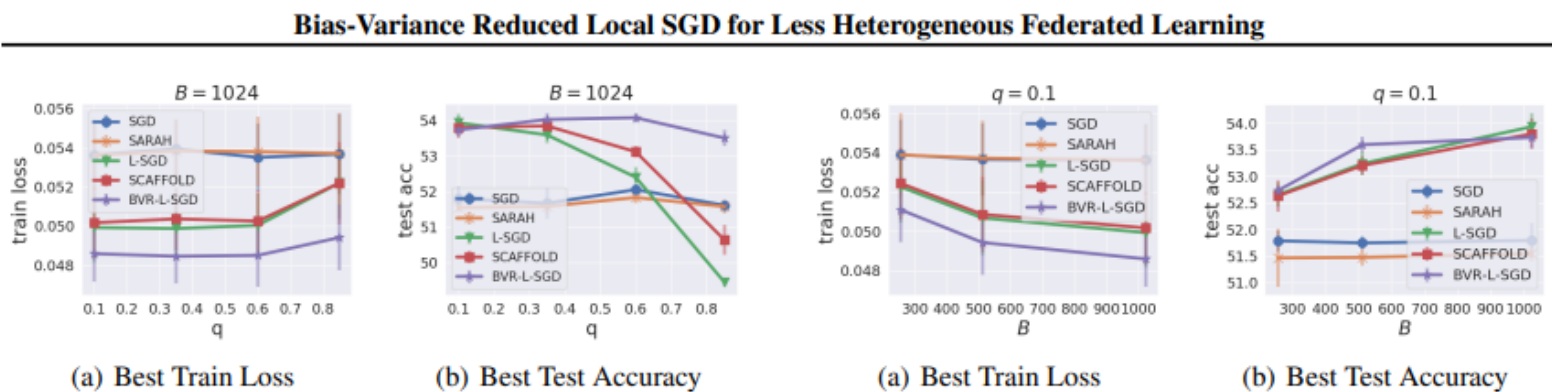


Figure 2. Comparison of the best train loss and test accuracy against heterogeneity parameter q .

Figure 3. Comparison of the best train loss and test accuracy against local computation budget B .

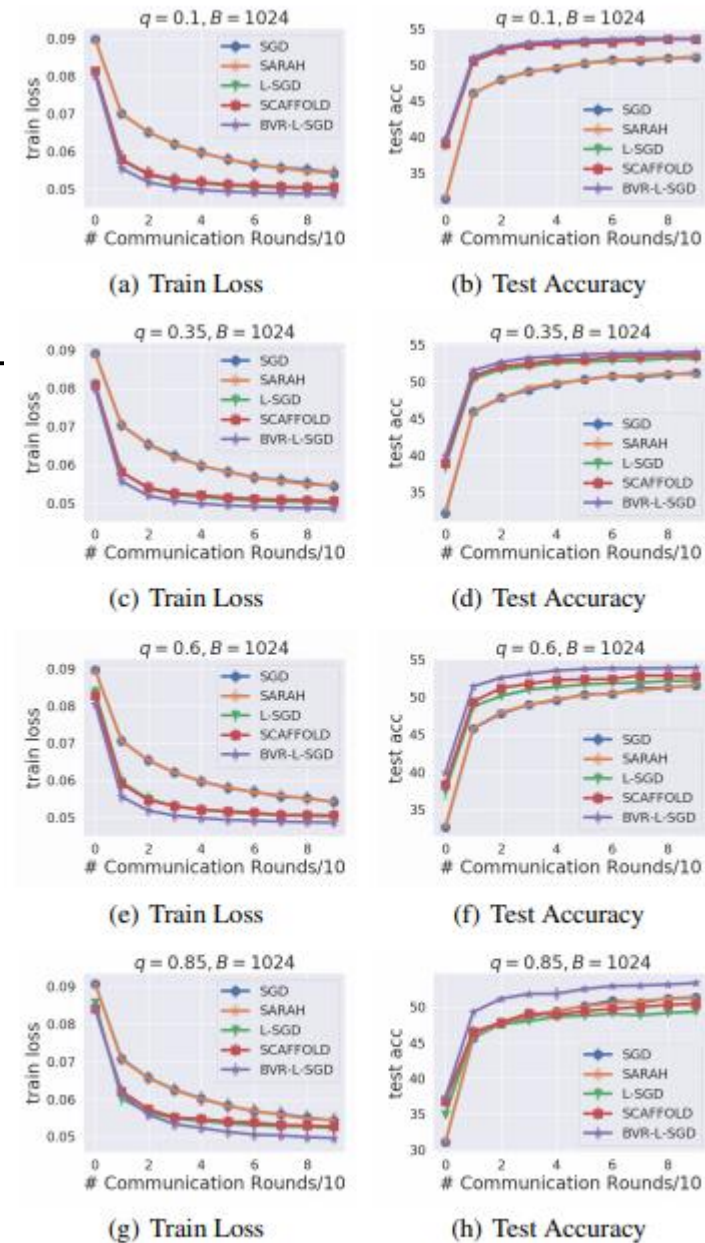


Figure 4. Comparison of the train loss and test accuracy against the number of communication rounds.

Contribution

- Nonconvex distributed learning에 사용하는 새로운 algorithm.
- BVR-L-SGD가 통신 복잡성 측면에서 이전의 local방법 뿐 아니라 non-local 방법보다 우수
- Second-order heterogeneity을 최대한 활용하기 위해 기존 로컬 방법에서 사용되는 평균을 취하는 방법 대신 무작위로 선택된 로컬 모델을 글로벌 모델로 사용
- 완화 조건인 heterogeneity가 클 때 기존 local sgd들의 성능은 크게 저하되는 반면, BVR-L-SGD는 그 정도가 훨씬 작음.
- 관심 Domain인 Big-data분야에서 deeplearning 작업시 optimization에서 sgd 대신 bvr-l-sgd를 사용하면, 더 낮은 complexity와 더 빠른 속도 그리고 정확도를 기대해 볼 수 있을 것.

패턴인식 기말발표

Learning Disentangled representation via product manifold projection

ICML 2021'

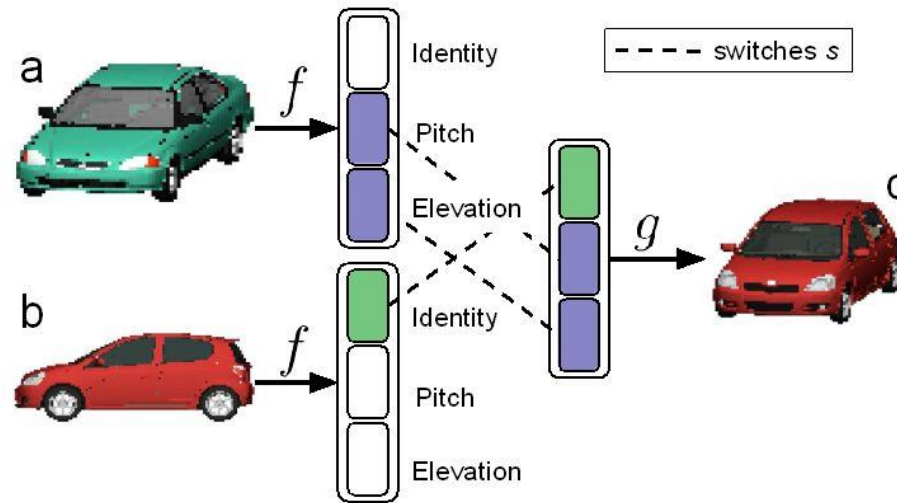
휴먼인터페이스 연구소

2021-25078 김지환

2021. 11. 20

Disentangle features

Learning a disentangled representation



$$\mathcal{L}_{dis} = \sum_{a,b,c,s \in \mathcal{D}} \|c - g(s \cdot f(a) + (1 - s) \cdot f(b))\|_2^2$$

©DeepAI

Contribution of the research

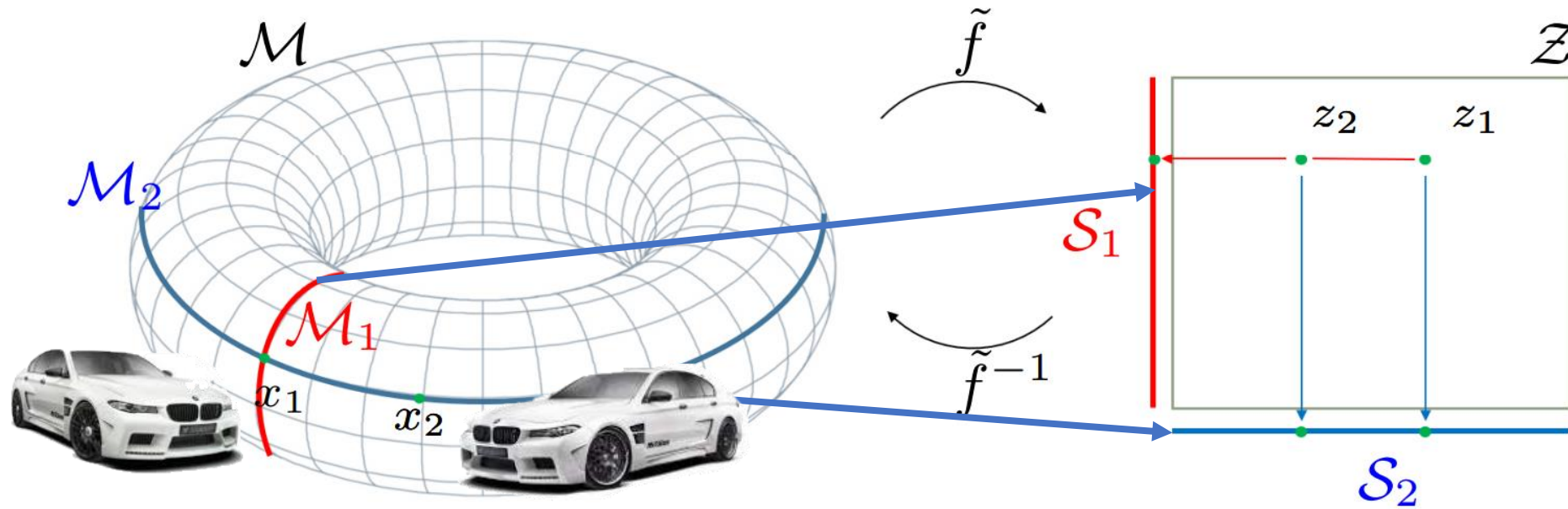
- 기하학적 관점에서 disentanglement를 해석 후, 이러한 이론적 토대가 현재 사용되는 방법론을 일반화할 수 있다는 것을 보임.
- Sample distribution의 관점에서, 기존 신경망 모델에 쉽게 적용가능한 weakly-supervised recipe를 제공
- 이러한 Recipe를 합성 데이터와 일반화하기 어려운 실제 데이터셋에 적용하고, 기존 방법론에 비해 더 나은 성능을 보임.

Theoretical justification

Basic Assumptions

- 높은 차원의 데이터는 낮은 차원의 Manifold에 자리한다
- 이 낮은 차원의 Manifold는 data의 **특정 feature를 modeling 하는 여러 개의 Submanifold의 product**로 이루어진다

Geometric definition of disentanglement

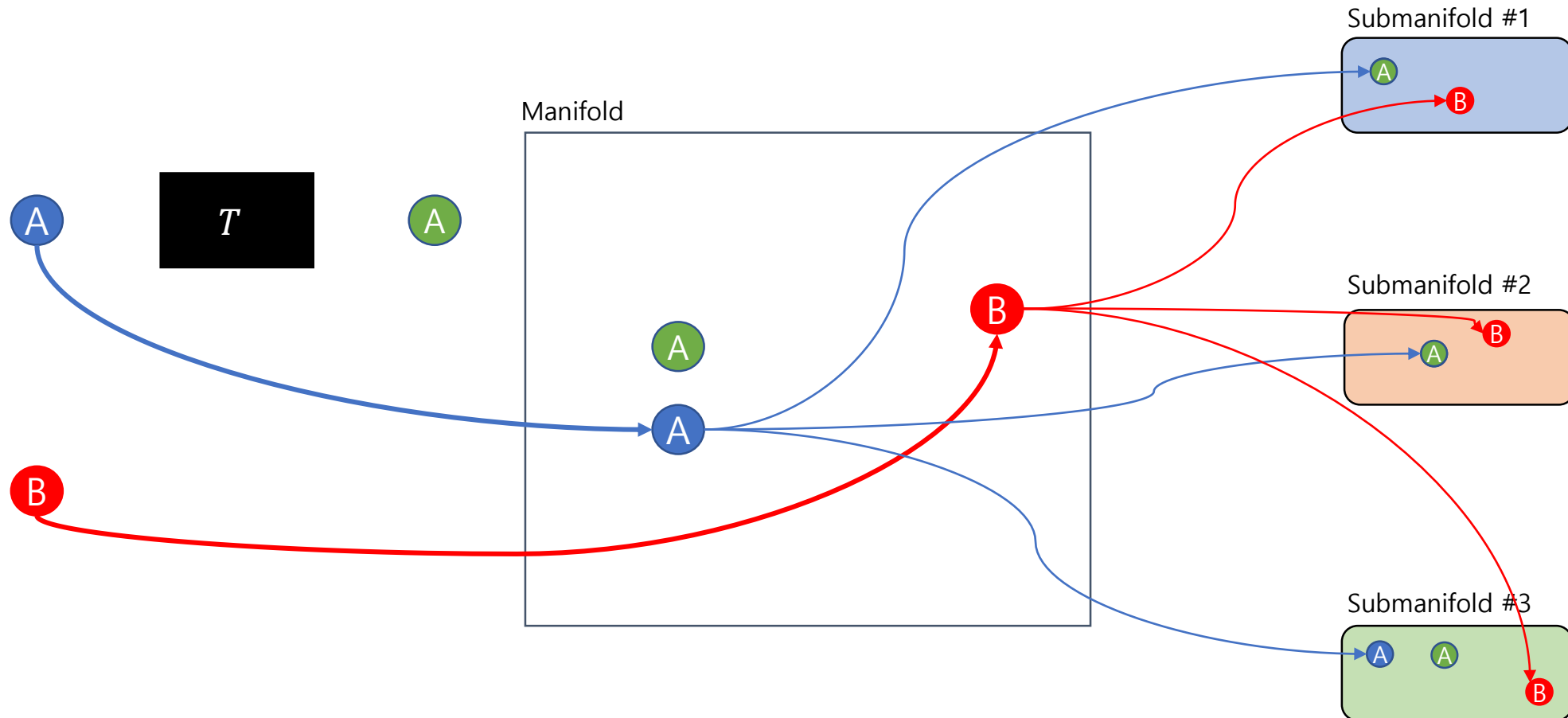


Definition 1 (Product metric space) A product metric space is an ordered pair (M, d) where M is a product of sets $M_1 \times \dots \times M_k$, each equipped with a function $d_i : M_i \times M_i \rightarrow \mathbb{R}$, $\forall i$ s.t. $\forall x, y, z \in M_i$:

$$\left[\begin{array}{l} d_i(x, y) = 0 \iff x = y \\ d_i(x, y) = d_i(y, x) \\ d_i(x, z) \leq d_i(x, y) + d_i(y, z) \end{array} \right.$$

The metric on the product space corresponds to the L_2 norm of the metric on the subspaces.

Disentangled representations

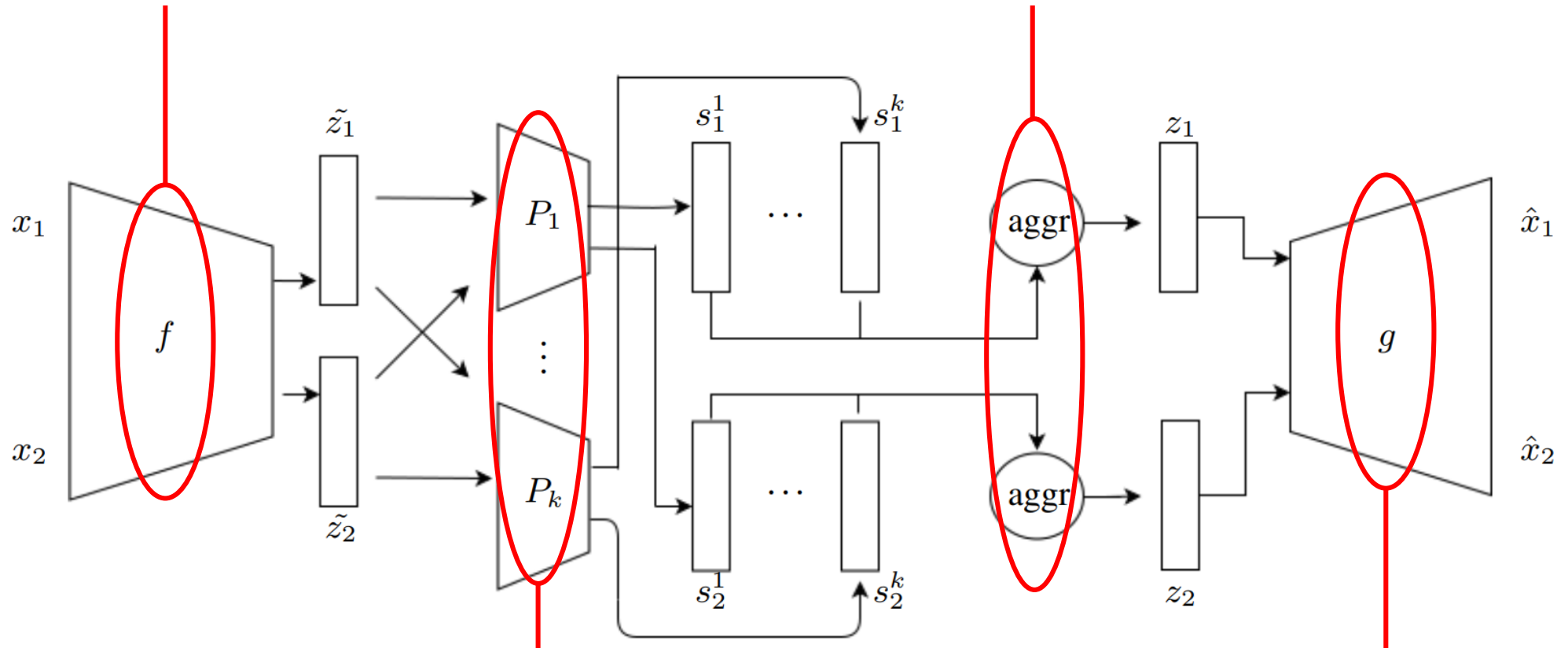


Methodology

Brief summary of the model

Embed data in pair to latent space

Sum all vectors in submanifolds



Reconstruct data sample

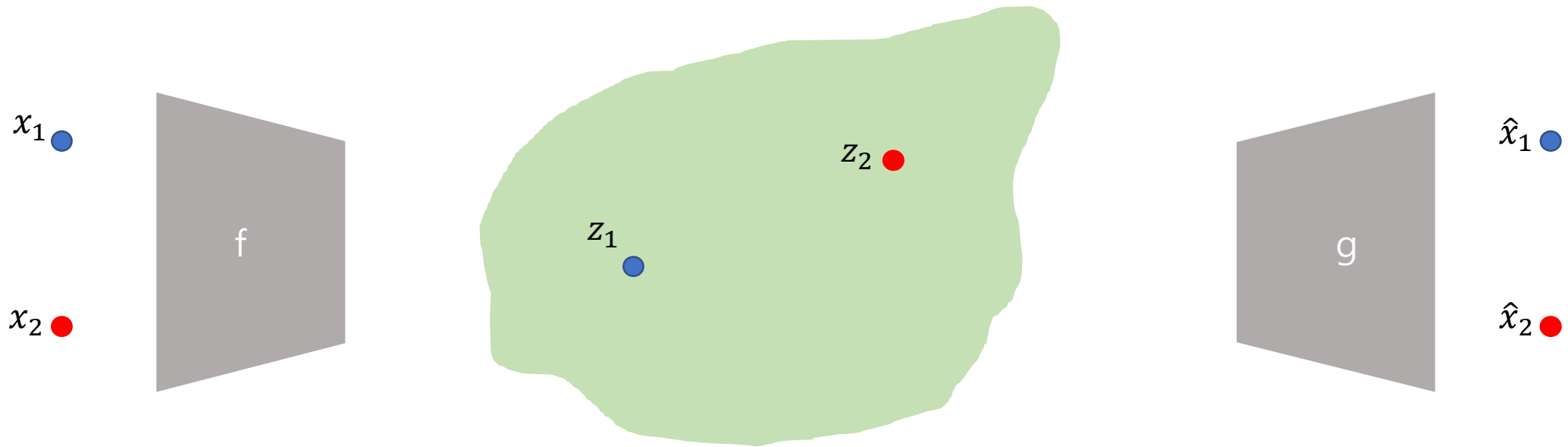
Embed latent to submanifold

Loss functions

$$\mathcal{L} = \mathcal{L}_{rec} + \beta_1(\mathcal{L}_{dis} + \mathcal{L}_{spar}) + \beta_2\mathcal{L}_{cons} + \beta_3\mathcal{L}_{reg}$$

Loss functions – reconstruction loss

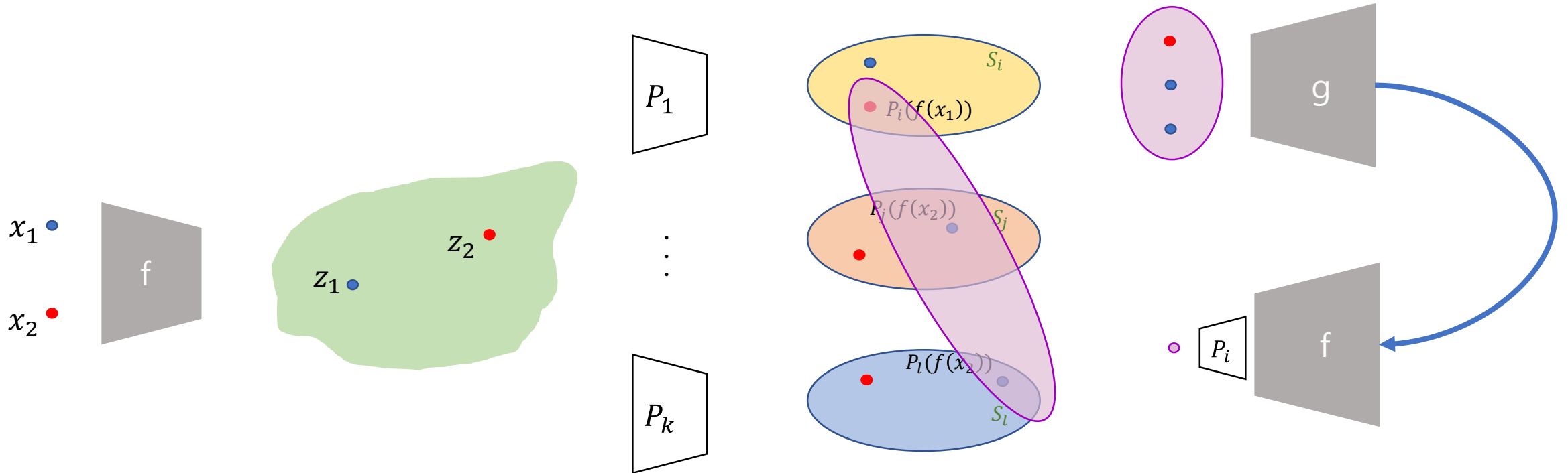
$$\mathcal{L}_{rec} = \|x - g_{\gamma}(\text{aggr}(P_{1,\omega_1} f_{\theta}(x), \dots, P_{k,\omega_k} f_{\theta}(x)))\|_2^2$$



Loss functions – consistency loss

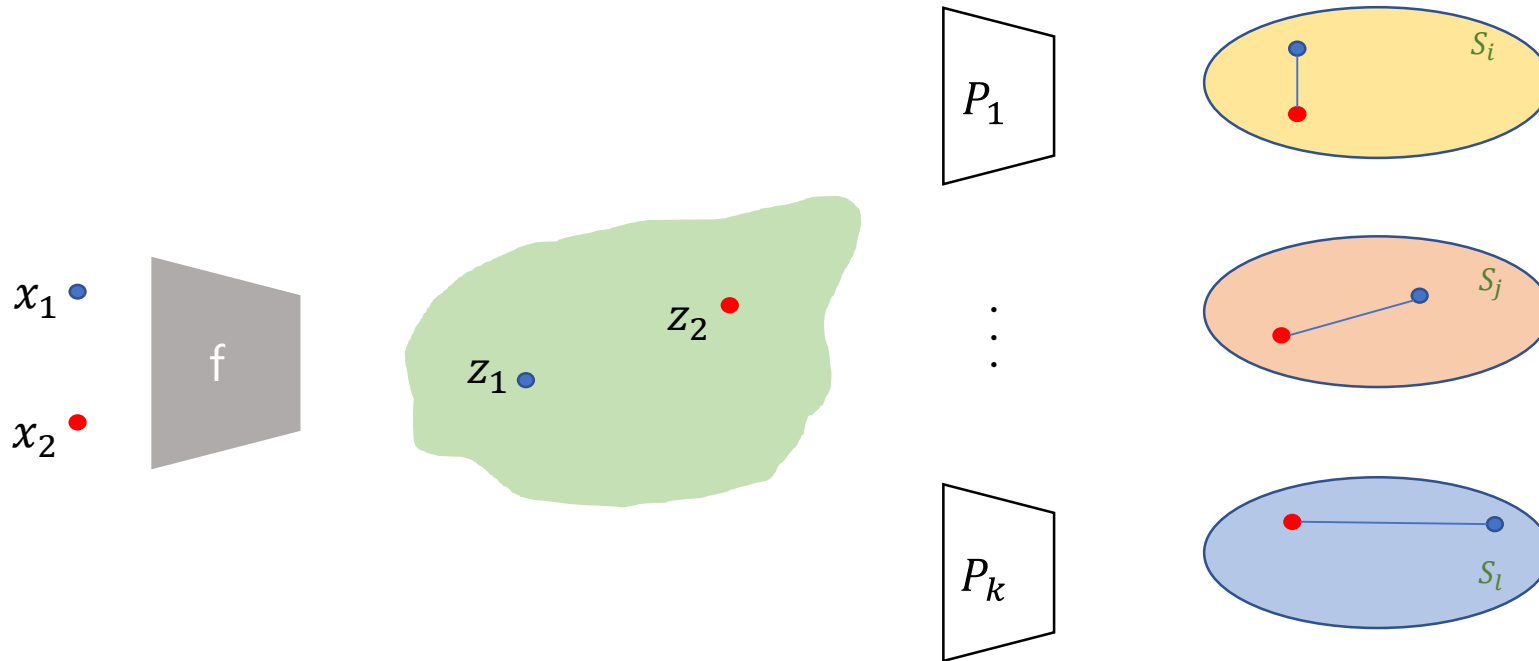
$$\mathcal{L}_{cons} = \sum_{i=1}^k \|P_{i,\omega_i}(f_{\theta}(\hat{x}_{s_i})) - s_i\|_2^2$$

$$s_i = P_i f(x_1) \text{ and } \hat{x}_{s_i} = g(\text{aggr}(P_i f(x_1), P_{j \neq i} f(x_2)))$$



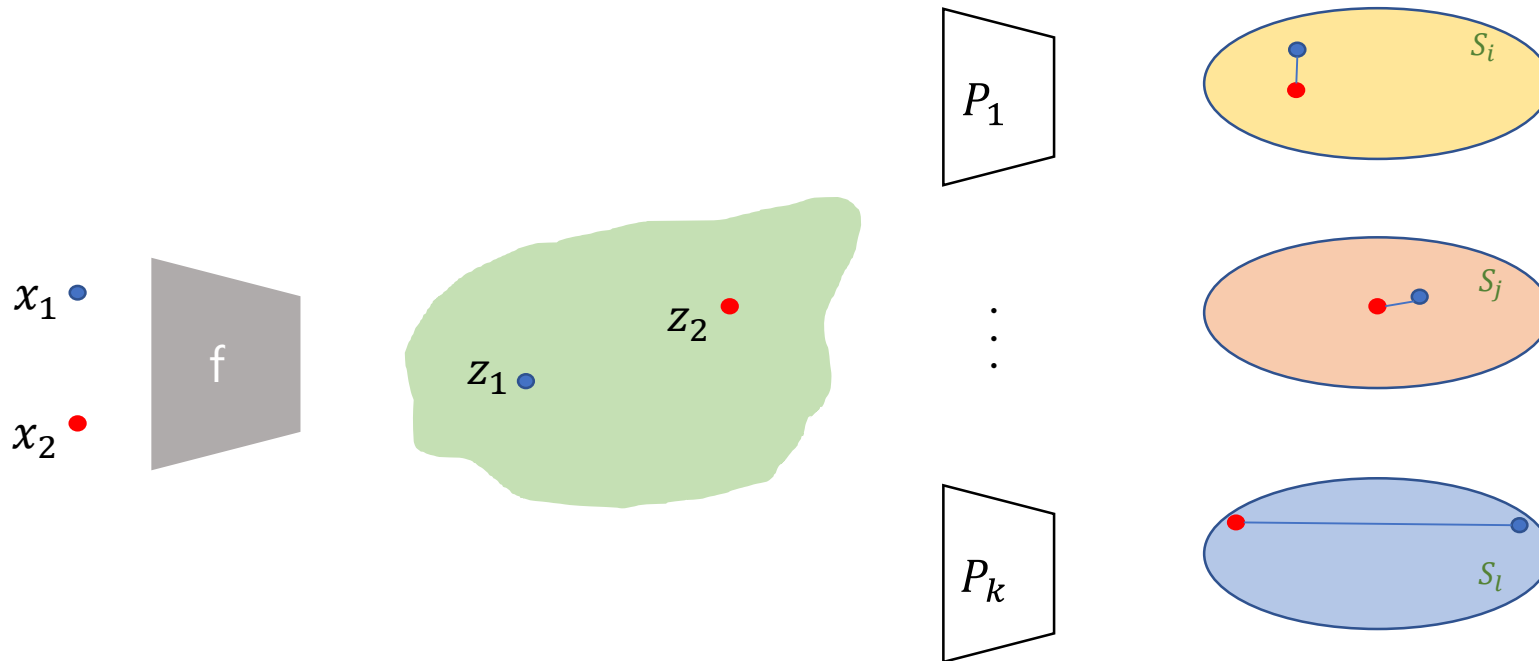
Loss functions – distance loss

$$\mathcal{L}_{dis} = \sum_{i=1}^k (1 - \alpha_i) \delta_i^2 + \alpha_i \max(m - \delta_i, 0)^2$$



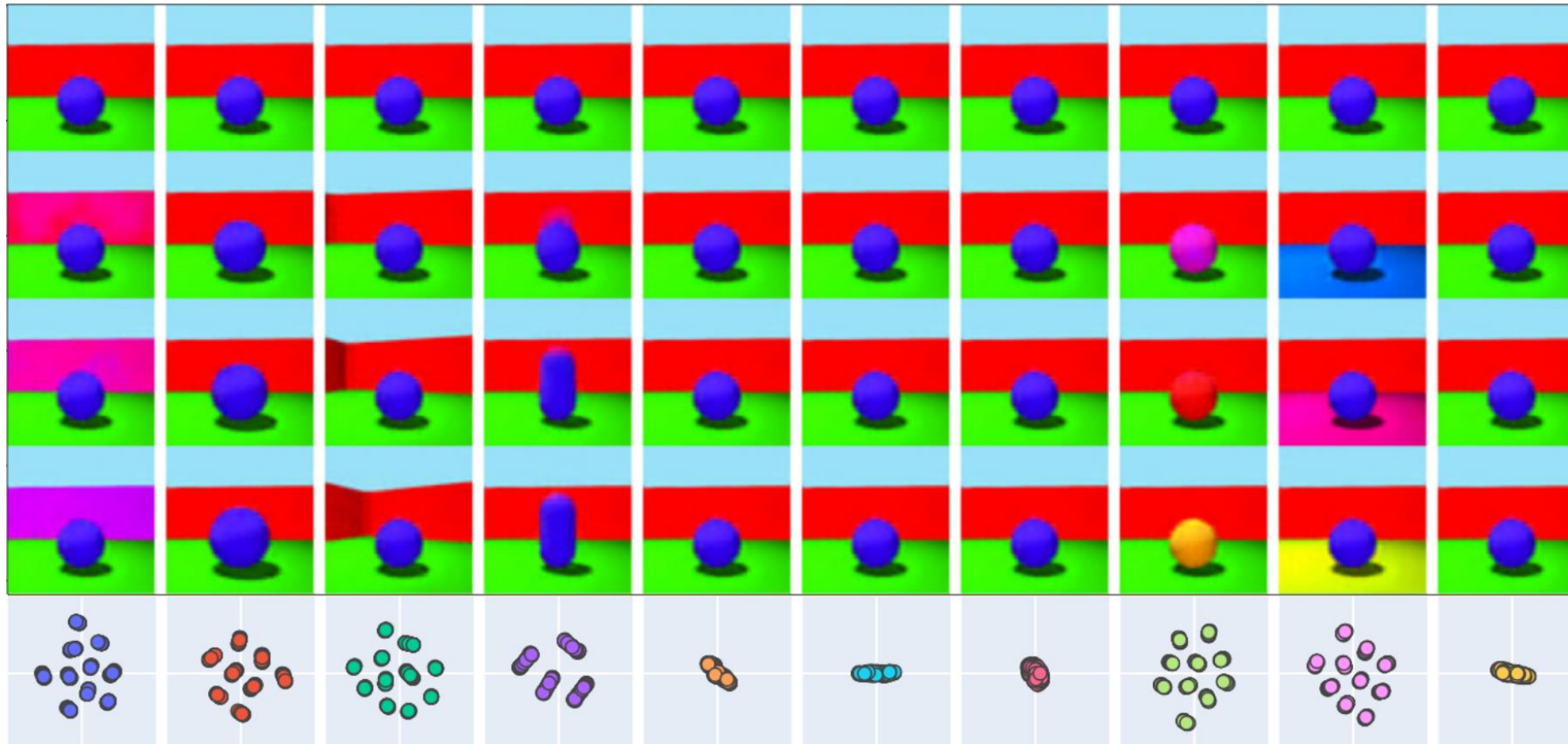
Loss functions – distance loss

$$\mathcal{L}_{dis} = \sum_{i=1}^k (1 - \alpha_i) \delta_i^2 + \alpha_i \max(m - \delta_i, 0)^2$$



Experimental result

Experimental result



Thank you



k-Nearest Neighbors by Means of Sequence to Sequence Deep Neural Networks and Memory Networks

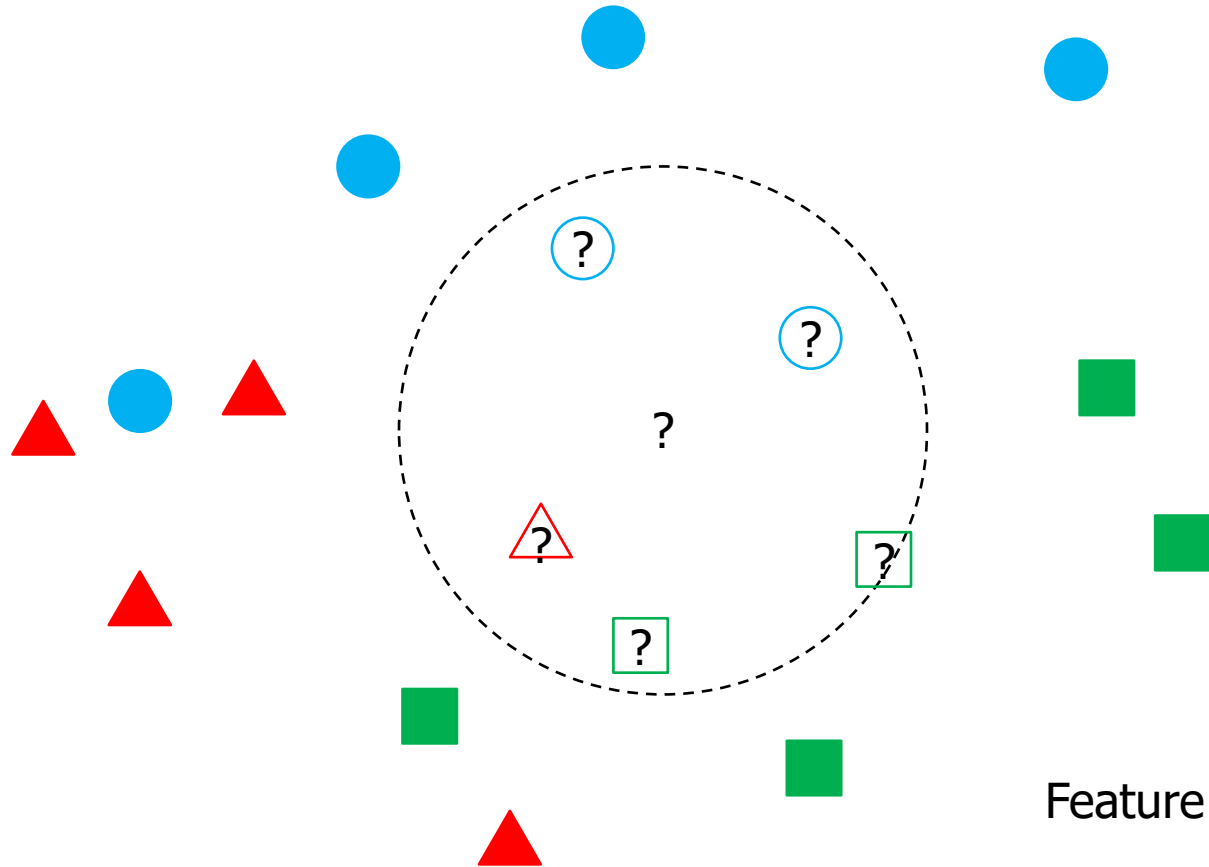
Yiming Xu and Diego Klabjan

Northwestern University

IJCAI 2021

Presenter: Jinwoo Kim

Introduction: Problem definition



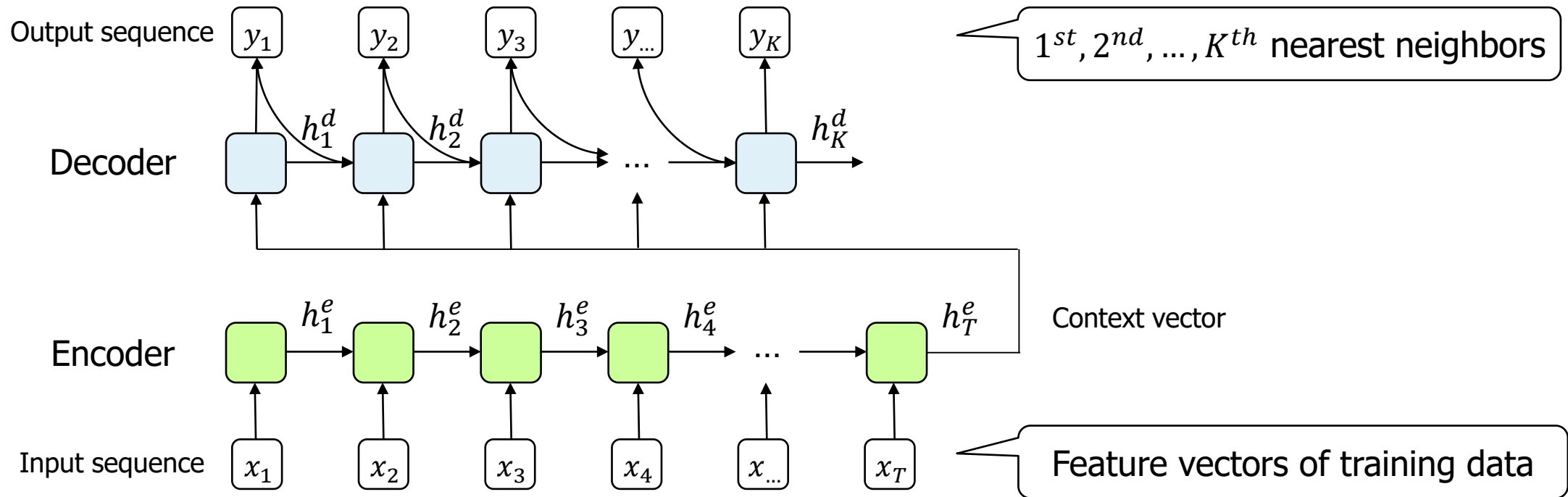
Task 1: Classification
Predict Label of test data,
Labels of K nearest neighbors,
Feature vectors of K nearest neighbors

Task 2: Oversampling
Add generated samples to minority class
for better classification

Feature space with data points with labels

Introduction: Sequence to Sequence Model

■ Architecture



Architecture of sequence to sequence model



Introduction:

kNN Models on Sequence to Sequence Model

- Below models are built on Seq2seq model:
 - Vector to label sequence (V2LS)
 - Vector to vector sequence (V2VS)
 - Vector to vector sequence and label sequence (V2VSLS)
- Notation
 - Input feature vector x
 - Feature vector X , label Y
 - Subscript $t = 1, \dots, K$: t^{th} nearest to x , superscript P, T : predicted, target
- Given
 - Input feature vector x and corresponding ground-truth label Y^{GT}
 - kNN upfront -> obtain labels $Y_1^T, Y_2^T, \dots, Y_K^T$ and feature vectors $X_1^T, X_2^T, \dots, X_K^T$

Sequence to Sequence Model: Vector to Label Sequence (V2LS)

- Predict label distributions Y_t^P of t^{th} nearest samples from x
 - $y_t \xrightarrow{\text{linear mapping}} W_y y_t + b_y \xrightarrow{\text{softmax with temperature } \tau} Y_t^P$
 - With τ lower than 1, Y_t^P becomes similar to one-hot vector
 - Preceding decoder cells preserve closeness to the original input, so $y_t \rightarrow t^{th}$ nearest
- Predict label distribution Y^P of x as:
 - $Y^P = \frac{1}{K} \sum_{t=1}^K Y_t^P$
 - Correspond to majority voting in kNN
- Loss function
 - $L_1 = E\left\{\frac{1}{K} \sum_{t=1}^K D_{KL}(Y_t^T || Y_t^P) + \alpha D_{KL}(Y^{GT} || Y^P)\right\}$



Sequence to Sequence Model: Vector to Vector Sequence (V2VS)

- Generate feature vector X_t^P corresponding to t^{th} nearest samples to x

- $y_t \xrightarrow{\text{linear mapping 1 with ReLU}} \sigma(W_{x1}y_t + b_{x1}) \xrightarrow{\text{linear mapping 2}} X_t^P$

- Loss function

- $L_2 = E\{\sum_{t=1}^K \|X_t^P - X_t^T\|^2\}$

- Objectives of generating out-of-sample feature vectors

- Make model to capture representative information of input and nearest neighbors
- Improve classification performance
- Enable oversampling



Sequence to Sequence Model: Vector to Vector Sequence and Label Sequence (V2VSLS)

- Joint training of V2VS and V2LS
 - Loss function $L = L_1 + \lambda L_2$
 - L_1 for learning both neighboring labels and the ground truth label
 - L_2 for learning neighboring vectors
 - Learn both label and feature vector of nearest neighbors, including label of x



Experimental Results

- Classification

	NI	COV	SensIT	CCD
XGB	87.53	91.98	82.56	66.95
FFN	88.53	91.83	83.67	65.37
V2VSLs	92.07	94.97	86.24	69.87
MNkNN_VEC	84.59	83.94	83.41	68.82

V2VSLs consistently outperforms the best classification models on all four datasets

F1 score comparison of full models

Experimental Results

■ Oversampling

Classification models	Oversampling Techniques	NI	COV	SensIT	CCD
FFN	No oversample	89.64	91.83	83.67	65.37
	SMOTE	89.99	91.18	83.43	66.32
	ADASYN	90.38	90.67	83.72	66.51
	V2VSL	90.89	92.05	83.94	66.82
XGB	No oversample	87.53	91.98	82.56	66.95
	SMOTE	87.79	91.86	82.87	66.56
	ADASYN	88.39	92.56	83.42	66.20
	V2VSL	87.62	92.43	82.46	66.96

K=5 out-of-sample
Add them to training set
until the classes are balanced

Oversampling: F1 score comparison

*Empirical Study of the Benefits of Overparameterization
in Learning Latent Variable Models*

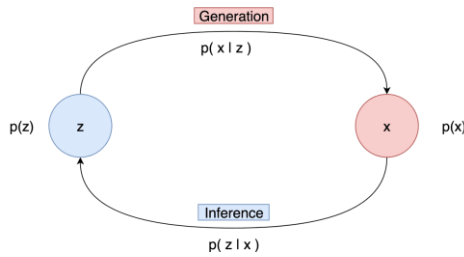
International Conference on Machine Learning (PMLR, 2020)

2021-27715
기계공학부

김태형

Latent Variable Model

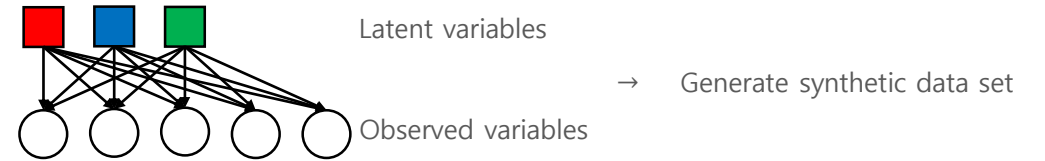
Latent variables : Variables that are not directly observed but are rather inferred from other variables that are observed (↔ Observable variable)



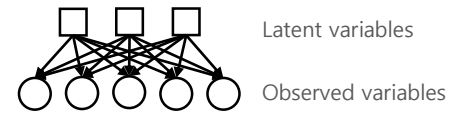
Overparameterization

having more model parameters than necessary to represent distribution of the data

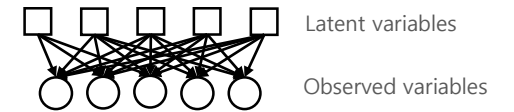
Ex) Ground truth model



Trained model



Non-overparameterized model

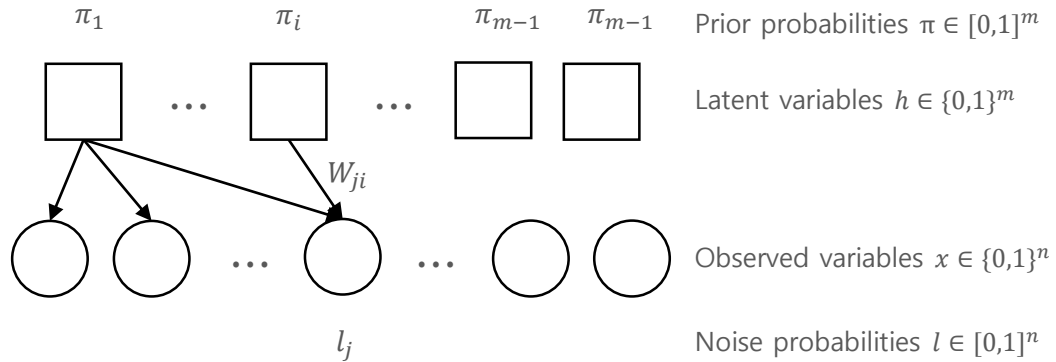


Overparameterized model

- Latent variable model을 학습하는 unsupervised learning에서 overparameterization의 영향에 대하여 empirical study
- 다양한 모델 및 학습 알고리즘(Noisy-OR network / sparse coding / probabilistic context-free grammars)에 대하여 진행
- Supervised learning에서는 overparameterization의 장점이 많이 연구됨

Noisy OR Network

- 1 binary latent variables layer & 1 binary observed variables layer
- Prior probabilities π , noise probabilities l



$\exp(-W_{ji})$: failure probability between related h_i and x_j

$$p(h) = \prod_{i=1}^m \pi_i^{h_i} (1 - \pi_i)^{1-h_i} \rightarrow p(x, h) = p(h) \prod_{j=1}^n p(x_j|h)$$

$$p(x_j = 0|h) = (1 - l_j) \prod_{i=1}^m \exp(-W_{ji} h_i)$$

Training algorithm

- $\log p(x; \theta) \geq E_{q(\cdot|x; \phi)} [\log p(x, h; \theta) - \log q(h|x; \phi)] = \mathcal{L}(x, \theta, \phi)$
- Assume $q(h|x; \phi) = \prod_{i=1}^m \sigma(x \cdot w_i + b_i)^{h_i} (1 - \sigma(x \cdot w_i + b_i))^{1-h_i}$
- Maximize the lower bound $\mathcal{L}(x, \theta, \phi)$ by taking gradient steps w.r.t. (θ, ϕ)

$q(h|x; \phi)$: variational posterior, also known as a recognition network

Extracting ground-truth latent variables

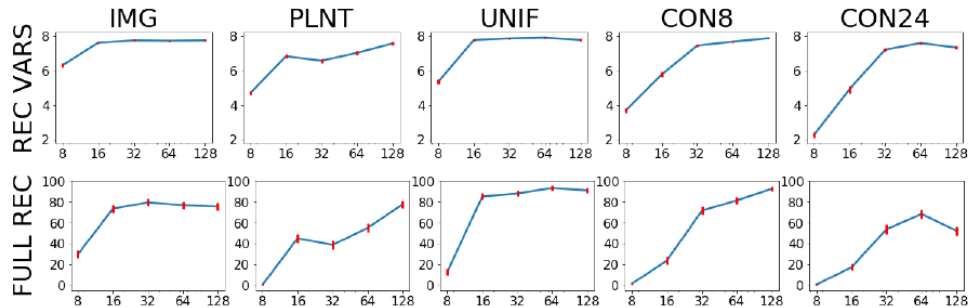
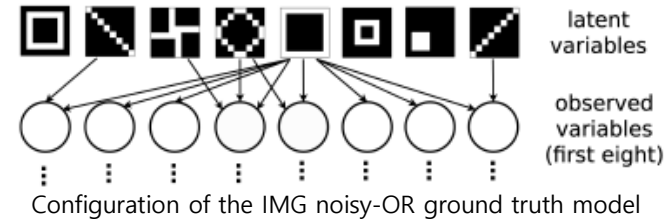
- $(\pi_i < 0.02)$ or $(\exp(-W_{ji}) > 0.8)$ for all related observable variables \Rightarrow discard latent variables
- Latent variables that are duplicates \Rightarrow discard the one with lower prior probability

<논문 제목> Empirical Study of the Benefits of Overparameterization in Learning Latent Variable Models

Results

Recovered* 되는 ground truth latent variables 수 측정

(Recovered : there exist a learned latent variable with the same parameters)



- y축 : 500번의 run 진행 후 recovered 된 latent variable 수

- y축 : percentage of runs with full ground truth recovery

- Larger model → recovered ground truth latent variables 증가
- Extreme Overparameterization의 단점 크게 나타나지 않음
- 학습 알고리즘이 바뀌어도 overparameterization의 긍정적인 영향에는 크게 변화가 없음



Contents lists available at [ScienceDirect](#)

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp



Multisource Domain Factorization Network for Cross-domain fault diagnosis of Rotating Machinery : An Unsupervised Multisource Domain Adaptation Method

Presenter: Taehun Kim

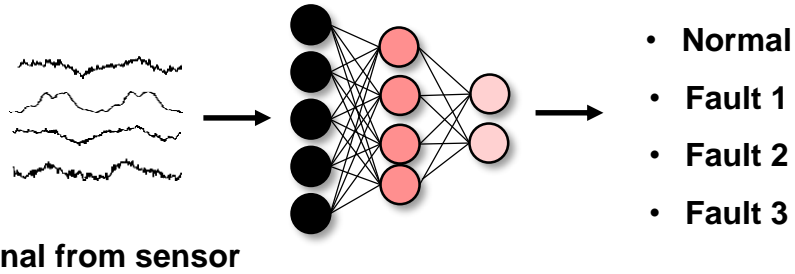
Department of Mechanical Engineering, Seoul National University

shrm.snu.ac.kr

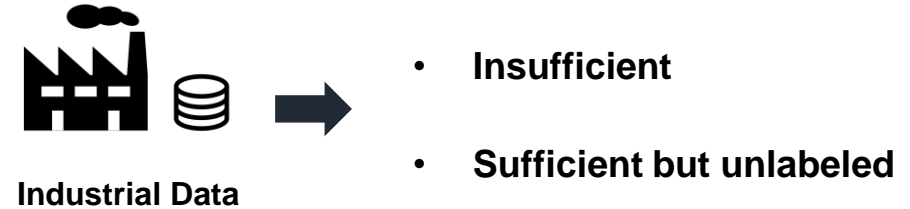
Room 215, Building 301, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul, 08826, Republic of Korea
Tel : +82-2-880-1664, +82-2-882-1664, +82-2-883-1664, +82-2-877-1884

Needs for Domain Adaptation in Fault Diagnosis

- **Deep-Learning based Fault Diagnosis**



- **Difficulties Obtaining Industrial Data**



- **Most Common Approaches**

- Training: several **labeled source domain** datasets and **unlabeled target domain** dataset
- Validation: **labeled target domain** dataset

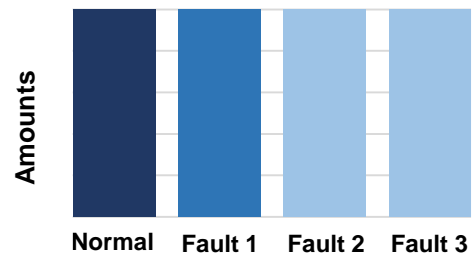
Source domain dataset

System



Testbed

Dataset



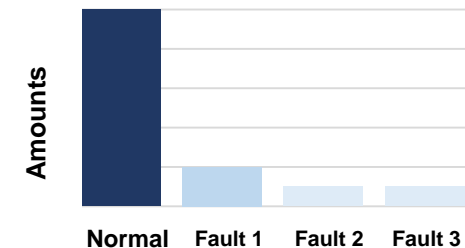
Target domain dataset

System



Industry

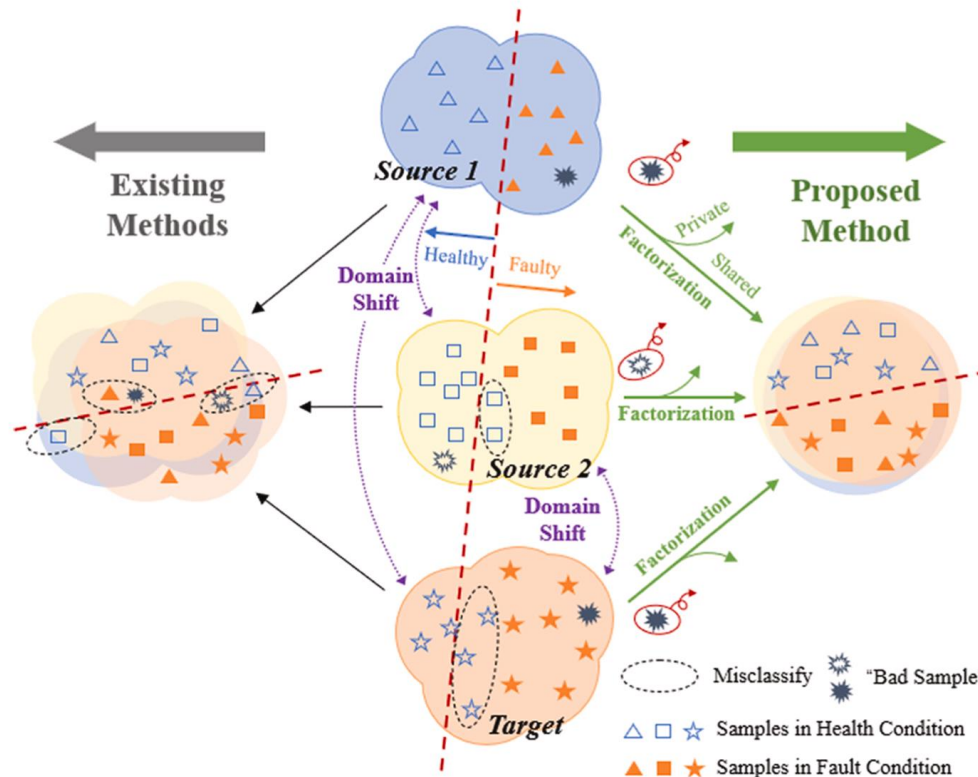
Dataset



Limitation of Conventional Domain Adaptation

- **Conventional Multisource Domain Adaptation**

- Purpose: Learn domain-invariant features



1) Vulnerable to noise

- Shared feature space is vulnerable to *noise
- Weaken the diagnostic performance

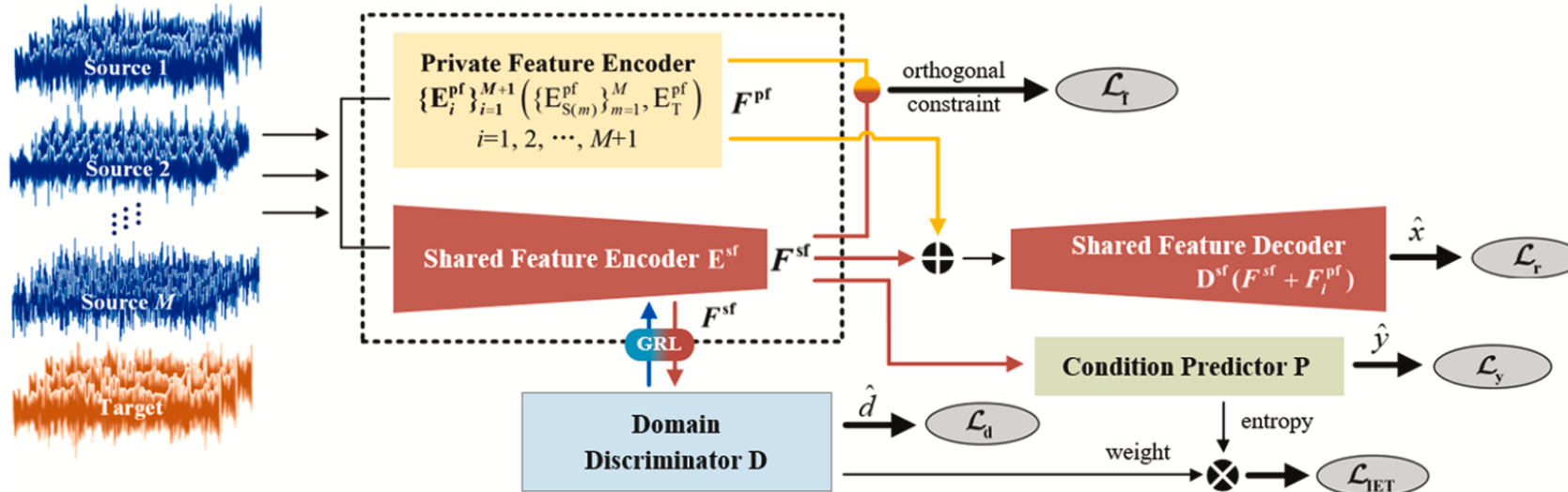
**Mechanical structure, working load, sampling frequency, and environmental noise*

2) Negative transfer issue

- Not all samples are transferable → "bad samples"
- Bad samples forcefully align the different domains
- It makes condition classifier confuse.

Proposed Method of MDFN

- Network of MDFN



1) Factorization loss

Squared Frobenius norm of source domain

$$\mathcal{L}_f = \sum_{m=1}^M \left(\left\| H_{S(m)}^{sfT} H_{S(m)}^{pf} \right\|_F^2 + \left\| H_T^{sfT} H_T^{pf} \right\|_F^2 \right)$$

Squared Frobenius norm of target domain

2) IET loss

Normalized domain discriminant entropy

$$\mathcal{L}_{IET} = \sum_{x_i \in D} w_i H(x_i) = - \sum_{i=1}^{N_{all}} \sum_{l=1}^L \left(1 - \frac{\sum_{m=1}^{M+1} \hat{d}_{i(m)} \log(\hat{d}_{i(m)})}{\log(M+1)} \right) p_{i(l)} \log(p_{i(l)})$$

Health condition entropy

THANK YOU



SHRM Laboratory for System Health and Risk Management
Department of Mechanical Engineering and Aerospace Engineering

Room 215, Building 301, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul, 08826, Republic of Korea
Tel : +82-2-880-1664, +82-2-882-1664, +82-2-883-1664, +82-2-877-1884

shrm.snu.ac.kr

Prototypical Contrastive Learning of

Prototypical Contrastive Learning of Unsupervised Representations

ICLR 2021

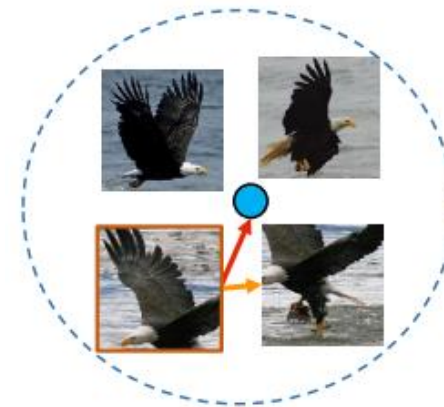
Hyungi Kim

패턴인식 기말논문발표

패턴인식 기말논문발표

Prototypical Contrastive Learning (PCL)

- Unsupervised (self-supervised) representation learning method
- Introduce prototypes as latent variables to help find the maximum-likelihood estimation of network parameters in an Expectation-Maximization (EM) framework
- Prototype is defined as “a representative embedding for a group of semantically similar instances”



Maximum Likelihood Estimation

- The objective is to find the network parameters θ that maximizes the log-likelihood function of the observed n samples

$$\theta^* = \arg \max_{\theta} \sum_{i=1}^n \log p(x_i; \theta)$$

- Assume that the observed data are related to latent variable $C = \{c_i\}_{i=1}^k$

- C denotes the prototypes of the data

- Can re-write the log-likelihood function as

$$\theta^* = \arg \max_{\theta} \sum_{i=1}^n \log p(x_i; \theta) = \arg \max_{\theta} \sum_{i=1}^n \log \sum_{c_i \in C} p(x_i, c_i; \theta)$$

Maximum Likelihood Estimation

$$\theta^* = \arg \max_{\theta} \sum_{i=1}^n \log p(x_i; \theta) = \arg \max_{\theta} \sum_{i=1}^n \log \sum_{c_i \in C} p(x_i, c_i; \theta)$$

- The log-likelihood can be expressed as

$$\sum_{i=1}^n \sum_{c_i \in C} Q(c_i) \log p(x_i, c_i; \theta)$$

$$Q(c_i) = \frac{p(x_i, c_i; \theta)}{\sum_{c_i \in C} p(x_i, c_i; \theta)} = \frac{p(x_i, c_i; \theta)}{p(x_i; \theta)} = p(c_i; x_i, \theta)$$

- Find θ^* via Expectation Maximization

E-step

- Aim to estimate $Q(c_i) = p(c_i; x_i, \theta)$
- Perform k-means on the features $v_i = f_\theta(x_i)$
- Define prototype c_i as the centroid for the i -th cluster
- $p(c_i; x_i, \theta) = \mathbb{1}(x_i \in c_i)$
 - $\mathbb{1}(x_i \in c_i) = 1$ if x_i belongs to the cluster represented by c_i
 - Otherwise 0

$$\sum_{i=1}^n \sum_{c_i \in C} Q(c_i) \log p(x_i, c_i; \theta)$$

M-step

$$\begin{aligned}\sum_{i=1}^n \sum_{c_i \in C} Q(c_i) \log p(x_i, c_i; \theta) &= \sum_{i=1}^n \sum_{c_i \in C} p(c_i; x_i, \theta) \log p(x_i, c_i; \theta) \\ &= \sum_{i=1}^n \sum_{c_i \in C} \mathbb{1}(x_i \in c_i) \log p(x_i, c_i; \theta)\end{aligned}$$

- Under the assumption of a uniform prior over cluster centroids

$$p(x_i, c_i; \theta) = p(x_i; c_i, \theta)p(c_i; \theta) = \frac{1}{k} \cdot p(x_i; c_i, \theta)$$

- Assume the distribution around each prototype c_i as an isotropic Gaussian

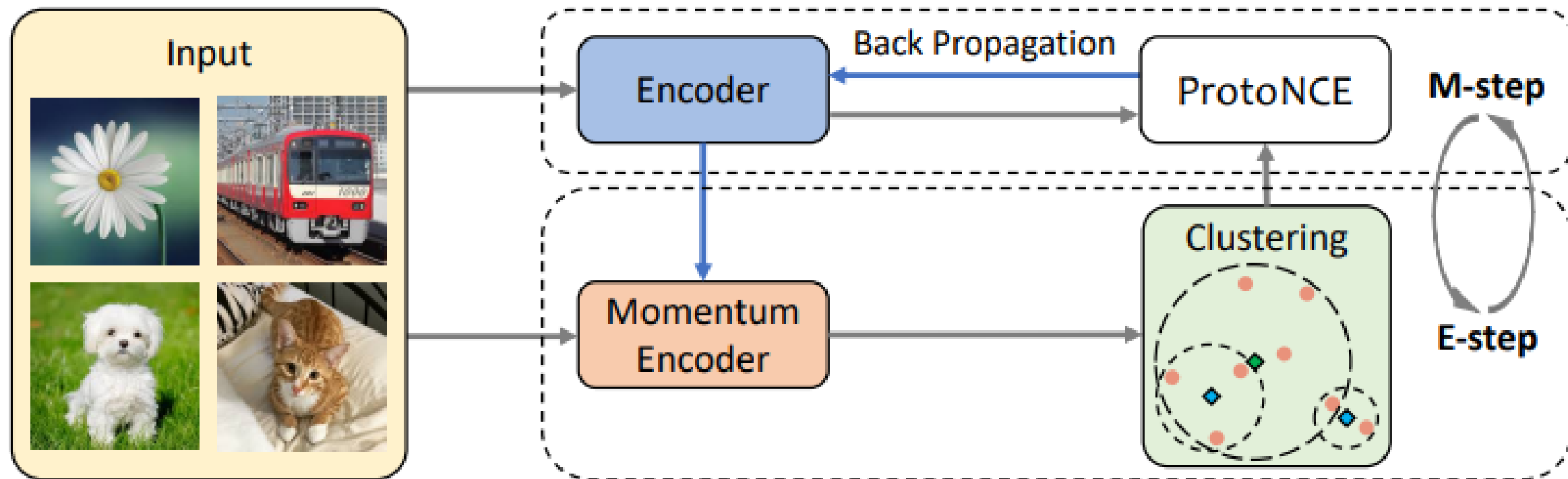
$$p(x_i; c_i, \theta) = \exp\left(\frac{-(v_i - c_s)^2}{2\sigma_s^2}\right) / \sum_{j=1}^k \exp\left(\frac{-(v_i - c_j)^2}{2\sigma_j^2}\right)$$

M-step

- If v and c are l_2 -normalized, $(v - c)^2 = 2 - 2v \cdot c$
- Rewrite maximum log-likelihood estimation as

$$\theta^* = \arg \min_{\theta} \sum_{i=1}^n -\log \frac{\exp(v_i \cdot c_s / \phi_s)}{\sum_{j=1}^k \exp(v_i \cdot c_j / \phi_j)}$$

Summary



Experiments

- Low-shot image classification

Method	architecture	VOC07					Places205				
		$k=1$	$k=2$	$k=4$	$k=8$	$k=16$	$k=1$	$k=2$	$k=4$	$k=8$	$k=16$
Random	ResNet-50	8.0	8.2	8.2	8.2	8.5	0.7	0.7	0.7	0.7	0.7
Supervised		54.3	67.8	73.9	79.6	82.3	14.9	21.0	26.9	32.1	36.0
Jigsaw	ResNet-50	26.5	31.1	40.0	46.7	51.8	4.6	6.4	9.4	12.9	17.4
MoCo		31.4	42.0	49.5	60.0	65.9	8.8	13.2	18.2	23.2	28.0
PCL (ours)		46.9	56.4	62.8	70.2	74.3	11.3	15.7	19.5	24.1	28.4
SimCLR		32.7	43.1	52.5	61.0	67.1	9.4	14.2	19.3	23.7	28.3
MoCo v2	ResNet-50-MLP	46.3	58.3	64.9	72.5	76.1	10.9	16.3	20.8	26.0	30.1
PCL v2 (ours)		47.9	59.6	66.2	74.5	78.3	12.5	17.5	23.2	28.1	32.3

Table 1: **Low-shot image classification** on both VOC07 and Places205 datasets using linear SVMs trained on fixed representations. All methods were pretrained on ImageNet-1M dataset for 200 epochs (except for Jigsaw trained on ImageNet-14M). We vary the number of labeled examples k and report the mAP (for VOC) and accuracy (for Places) across 5 runs. We use the released pretrained model for MoCo, and re-implement SimCLR.

Experiments

- Semi-supervised learning

Method	architecture	#pretrain epochs	Top-5 Accuracy	
			1%	10%
Random (Wu et al., 2018)	ResNet-50	-	22.0	59.0
Supervised baseline (Zhai et al., 2019)	ResNet-50	-	48.4	80.4
<i>Semi-supervised learning methods:</i>				
Pseudolabels (Zhai et al., 2019)	ResNet-50v2	-	51.6	82.4
VAT + Entropy Min. (Miyato et al., 2019)	ResNet-50v2	-	47.0	83.4
S ⁴ L Rotation (Zhai et al., 2019)	ResNet-50v2	-	53.4	83.8
<i>Self-supervised learning methods:</i>				
Instance Discrimination (Wu et al., 2018)	ResNet-50	200	39.2	77.4
Jigsaw (Noroozi & Favaro, 2016)	ResNet-50	90	45.3	79.3
SimCLR (Chen et al., 2020a)	ResNet-50-MLP	200	56.5	82.7
MoCo (He et al., 2020)	ResNet-50	200	56.9	83.0
MoCo v2 (Chen et al., 2020b)	ResNet-50-MLP	200	66.3	84.4
PCL v2 (ours)	ResNet-50-MLP	200	73.9	85.0
PCL (ours)	ResNet-50	200	75.3	85.6
PIRL (Misra & van der Maaten, 2020)	ResNet-50	800	57.2	83.8
SimCLR Chen et al. (2020a)	ResNet-50-MLP	1000	75.5 [†]	87.8 [†]
BYOL (Grill et al., 2020)	ResNet-50-MLP _{big}	1000	78.4 [†]	89.0 [†]
SwAV (Caron et al., 2020)	ResNet-50-MLP	800	78.5 [‡]	89.9 [‡]



Contrastive Learning

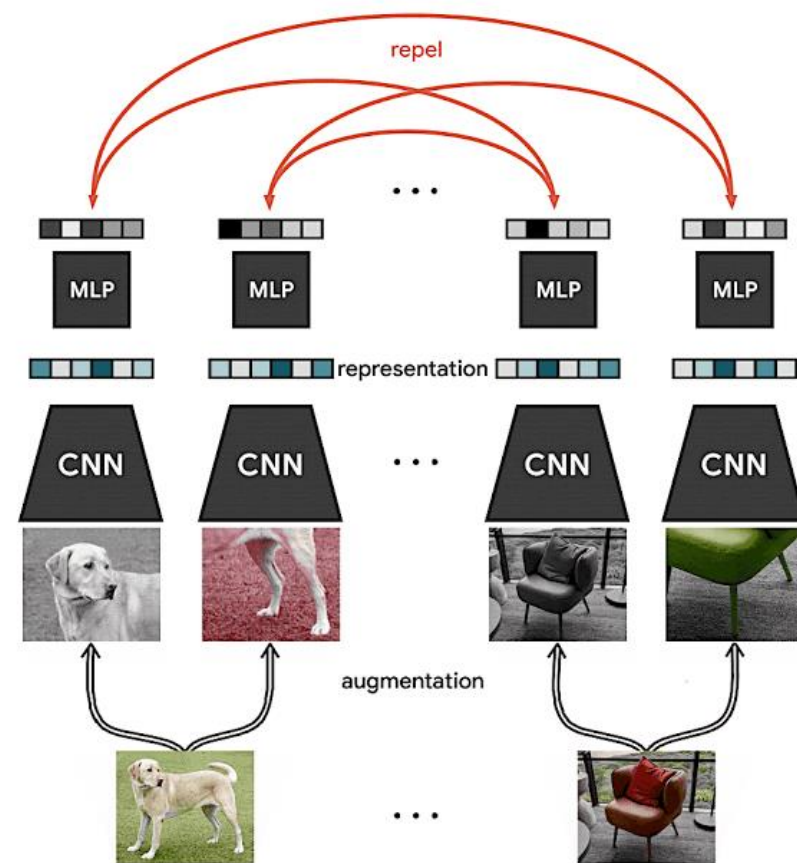
- Unsupervised (self-supervised) representation learning method

- One successful use case of contrastive loss

- Positive sample
- Negative samples

- $X = \{x_1, x_2, \dots, x_n\}$
- Embedding function f_θ
- $v_i = f_\theta(x_i)$

$$\mathcal{L}_{\text{InfoNCE}} = \sum_{i=1}^n -\log \frac{\exp(v_i \cdot v'_i / \tau)}{\sum_{j=0}^r \exp(v_i \cdot v'_j / \tau)}$$



Maximum Likelihood Estimation

- It is hard to optimize this function directly
 - Use a surrogate function Q to lower-bound
 - $\sum_{c_i} Q(c_i) = 1$

$$\theta^* = \arg \max_{\theta} \sum_{i=1}^n \log p(x_i; \theta) = \arg \max_{\theta} \sum_{i=1}^n \log \sum_{c_i \in C} p(x_i, c_i; \theta)$$

$$\sum_{i=1}^n \log \sum_{c_i \in C} p(x_i, c_i; \theta) = \sum_{i=1}^n \log \sum_{c_i \in C} Q(c_i) \frac{p(x_i, c_i; \theta)}{Q(c_i)} \geq \sum_{i=1}^n \sum_{c_i \in C} Q(c_i) \log \frac{p(x_i, c_i; \theta)}{Q(c_i)}$$

Jensen's inequality

Maximum Likelihood Estimation

$$\sum_{i=1}^n \log \sum_{c_i \in C} p(x_i, c_i; \theta) = \sum_{i=1}^n \log \sum_{c_i \in C} Q(c_i) \frac{p(x_i, c_i; \theta)}{Q(c_i)} \geq \sum_{i=1}^n \sum_{c_i \in C} Q(c_i) \log \frac{p(x_i, c_i; \theta)}{Q(c_i)}$$

- To make the inequality hold with equality, $\frac{p(x_i, c_i; \theta)}{Q(c_i)}$ should be a constant

$$Q(c_i) = \frac{p(x_i, c_i; \theta)}{\sum_{c_i \in C} p(x_i, c_i; \theta)} = \frac{p(x_i, c_i; \theta)}{p(x_i; \theta)} = p(c_i; x_i, \theta)$$

- By ignoring the constant $-\sum_{i=1}^n \sum_{c_i \in C} Q(c_i) \log Q(c_i)$, one should maximize

$$\sum_{i=1}^n \sum_{c_i \in C} Q(c_i) \log p(x_i, c_i; \theta)$$

- Find θ^* via Expectation Maximization

Image Restoration for Under-Display Camera

CVPR 2021

전기정보공학부 2019-28250

남승우



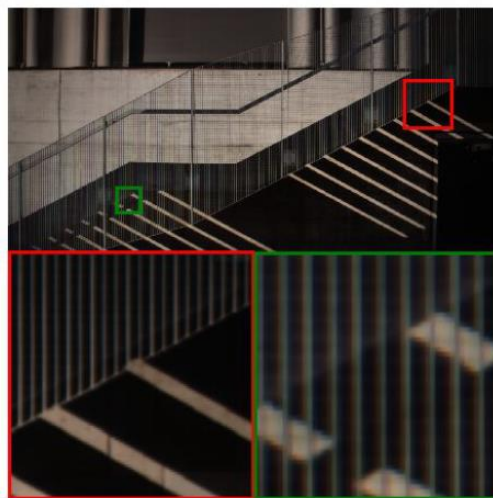
*Optical Engineering and Quantum Electronics Lab.,
Electrical and Computer Engineering,
Seoul National University, Republic of Korea*



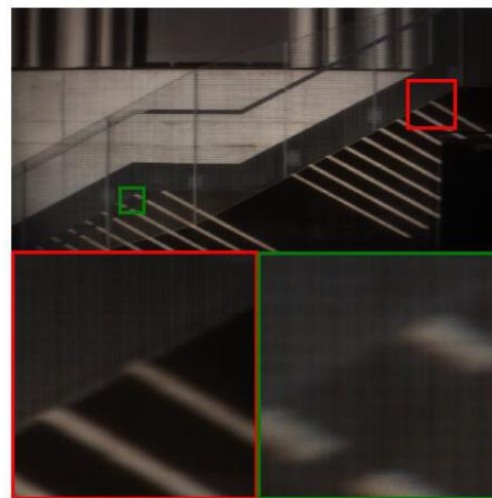
Under-Display Camera

- Under-Display Camera (UDC)
 - ▶ 디스플레이 아래에 카메라 존재
 - ▶ 노치 / 펀치홀 없는 넓은 디스플레이 구현
- Image degradation of UDC
 - ▶ 패널을 통과한 영상은 필연적으로 품질 저하

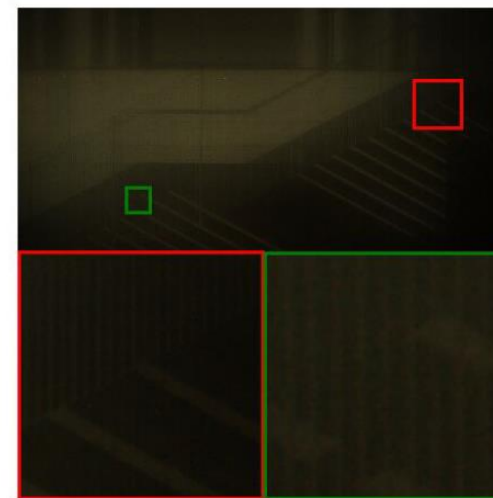
→ UDC 이미지를 복원하는 기술 필요



(a) Display-free



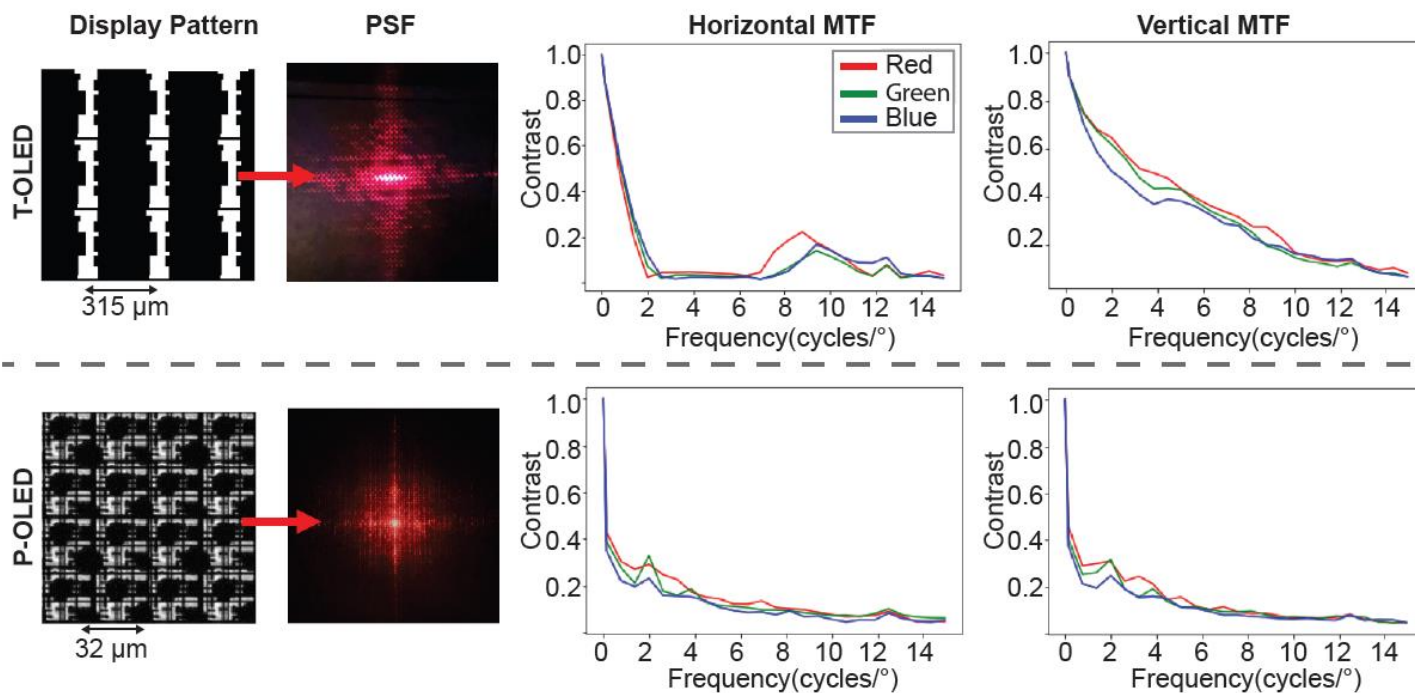
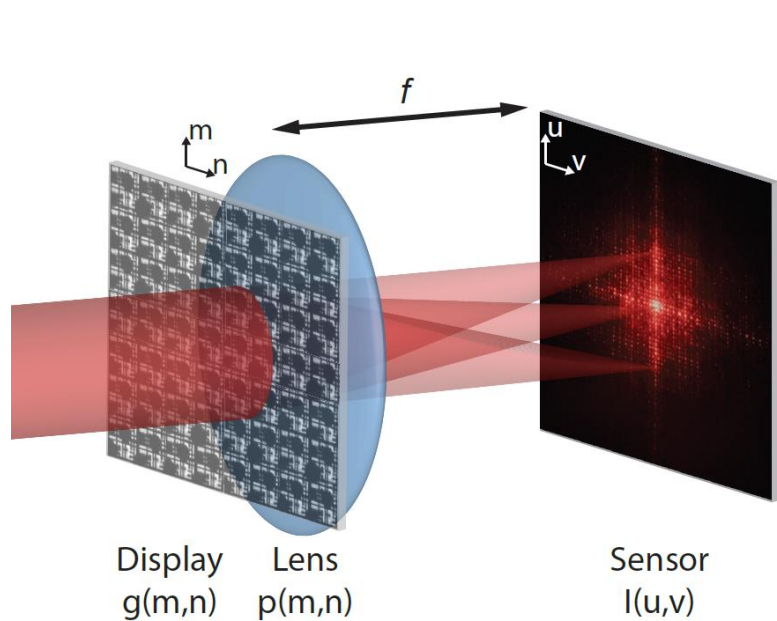
(b) TOLED



(c) POLED



Optical System analysis

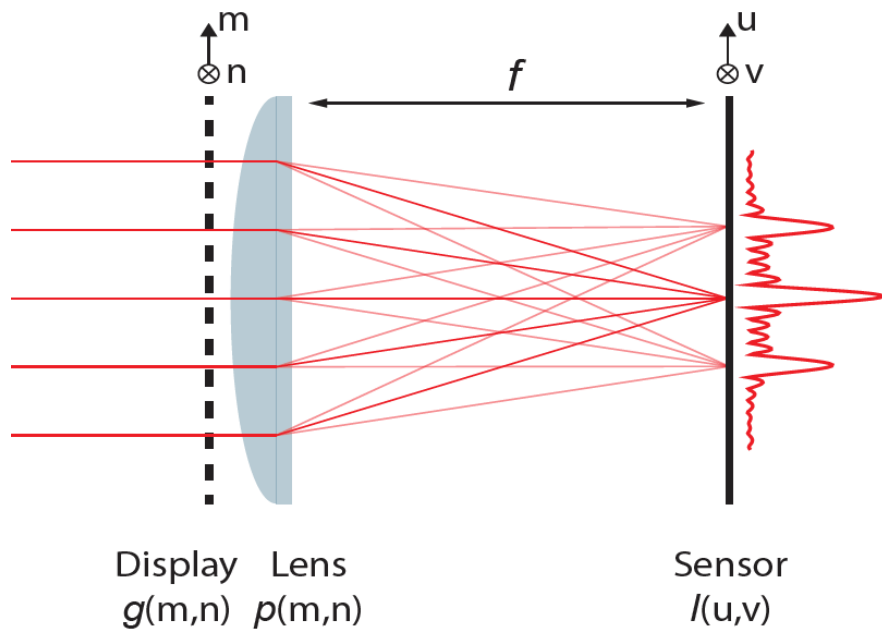


(-) ← $z = 0$ → (+)

Diffraction-limited
Point spread function (PSF)



Image formation & restoration



- Observation

$$\mathbf{y} = (\gamma \mathbf{x}) \otimes \mathbf{k} + \mathbf{n}, \quad \gamma = \frac{\int_S I_d(\delta, s) ds}{\int_S I_{nd}(0, s) ds}$$

- Point spread function

$$I(u, v) \propto \left| \iint_{-\infty}^{\infty} g_p(m, n) \exp \left[-j \frac{2\pi}{\lambda f} (mu + nv) \right] dm dn \right|^2$$

$$I(u, v) \propto |G_p(v_m, v_n)|^2 = \left| G_p \left(\frac{u}{\lambda f}, \frac{v}{\lambda f} \right) \right|^2$$

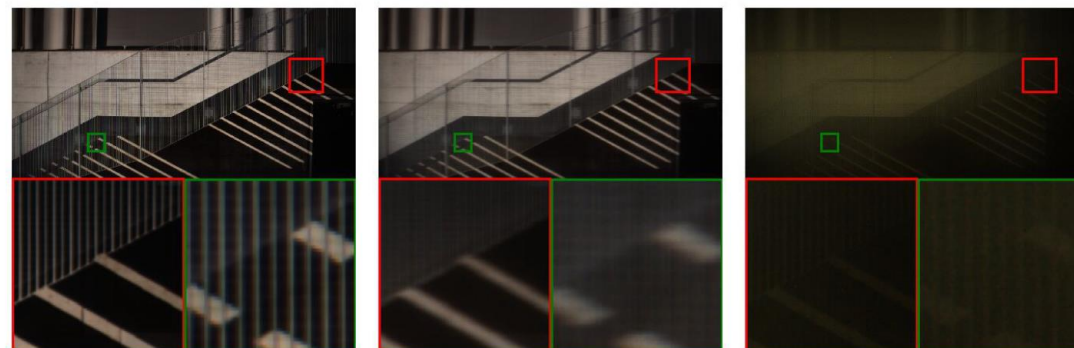
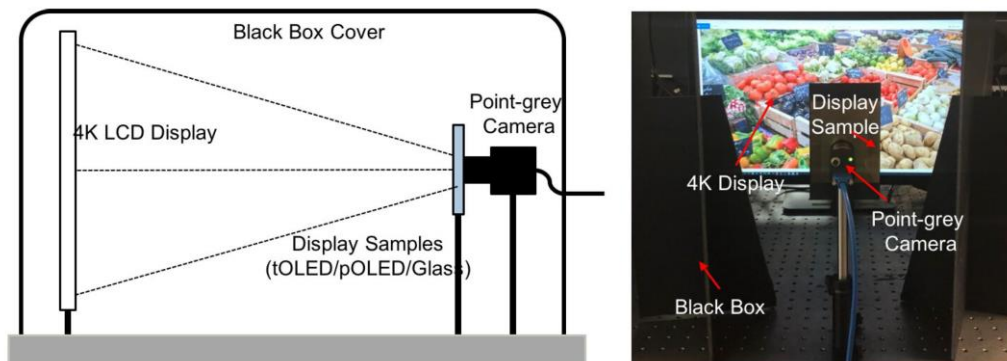
- Shot & readout noise

$$\mathbf{n} \sim \mathcal{N}(\mu = 0, \sigma^2 = \lambda_{read} + \lambda_{shot} w)$$



Learning-based methods

- Monitor-Camera Imaging System for dataset

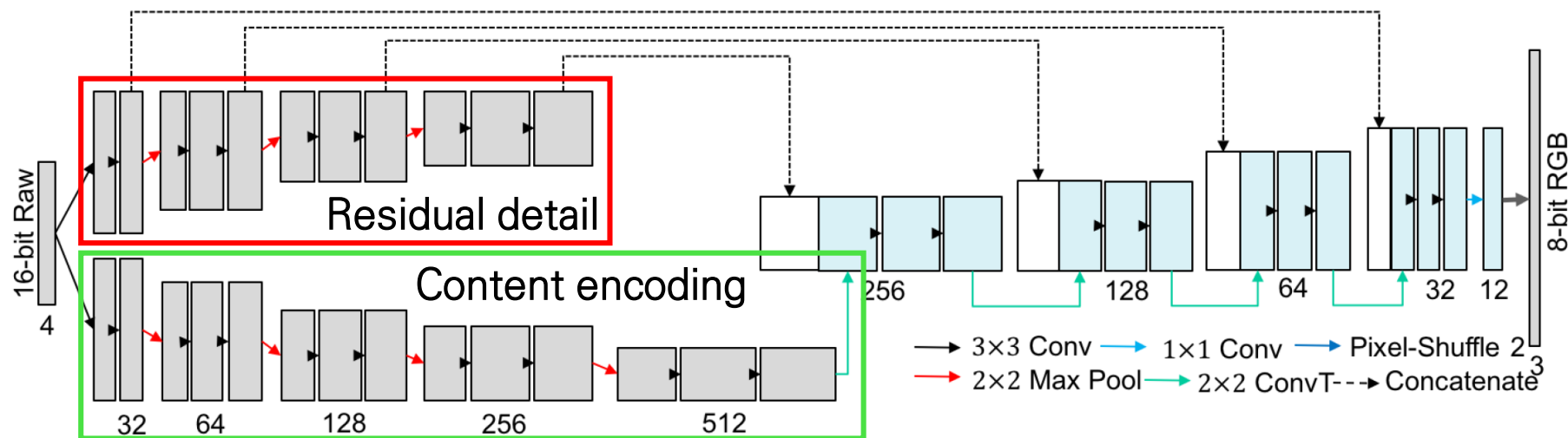


(a) Display-free

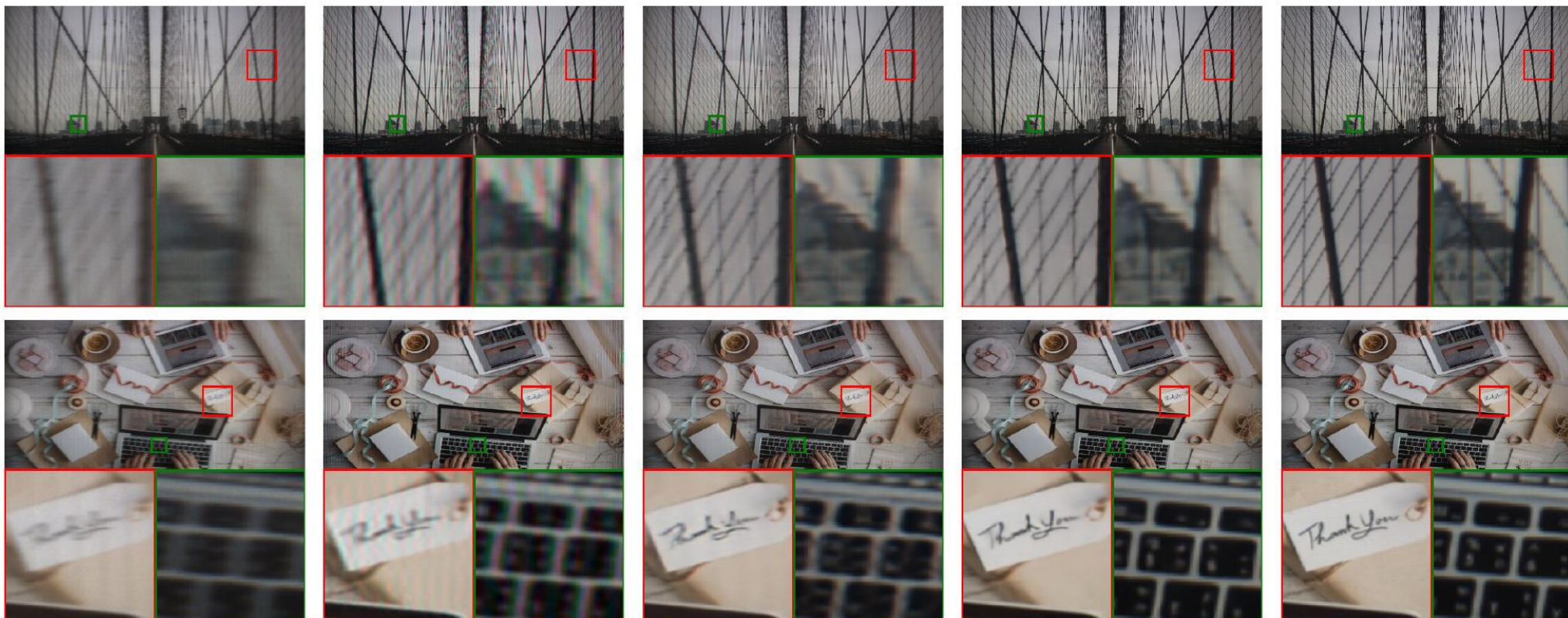
(b) TOLED

(c) POLED

- Proposed UNet structure



Results



(a) T-OLED

(b) DeP

(c) UNet(Syn)

(d) UNet

(e) GT

Table 4: Pipeline Comparison

				4K T-OLED		P-OLED	
Pipeline Structure	#P ↓	GFLOPs ↓	T ↓	PSNR/SSIM ↑	LPIPS ↓	PSNR/SSIM ↑	LPIPS ↓
DeP	-	-	-	28.50/0.9117	0.4219	16.97/0.7084	0.6306
ResNet	1.37M	721.76	92.92	36.26/0.9703	0.1214	27.42/0.9176	0.2500
UNet(Syn)	8.93M	124.36	21.37	32.42/0.9343	0.1739	25.88/0.9006	0.3089
UNet	8.93M	124.36	21.37	36.71/0.9713	0.1209	30.45/0.9427	0.2219



Results

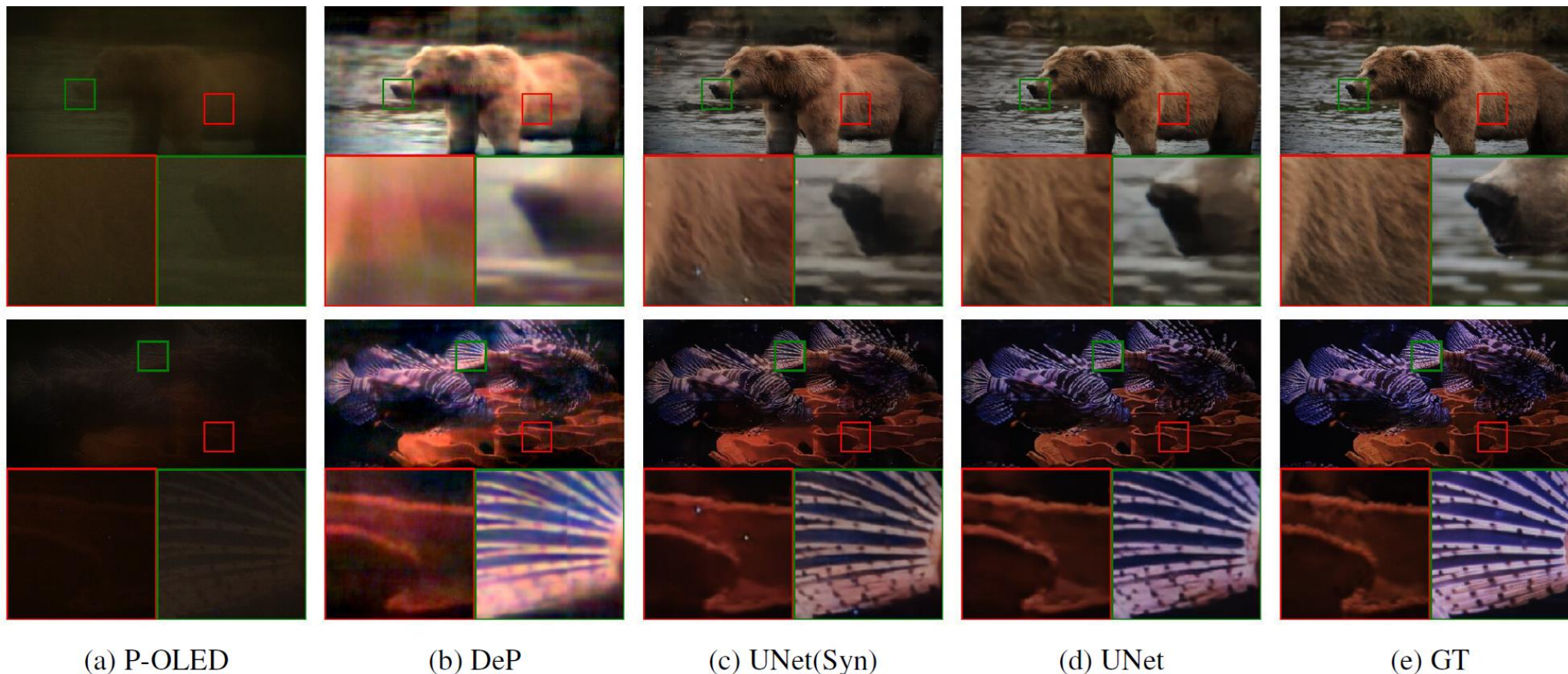


Table 4: Pipeline Comparison

Pipeline Structure	#P ↓	GFLOPs ↓	T ↓	4K T-OLED		P-OLED	
				PSNR/SSIM ↑	LPIPS ↓	PSNR/SSIM ↑	LPIPS ↓
DeP	-	-	-	28.50/0.9117	0.4219	16.97/0.7084	0.6306
ResNet	1.37M	721.76	92.92	36.26/0.9703	0.1214	27.42/0.9176	0.2500
UNet(Syn)	8.93M	124.36	21.37	32.42/0.9343	0.1739	25.88/0.9006	0.3089
UNet	8.93M	124.36	21.37	36.71/0.9713	0.1209	30.45/0.9427	0.2219



Thank You



ReLIC

Representation Learning via Invariant Causal Mechanisms

Published in ICLR 2021

211120

Seongjun Mun

sjmun@cml.snu.ac.kr

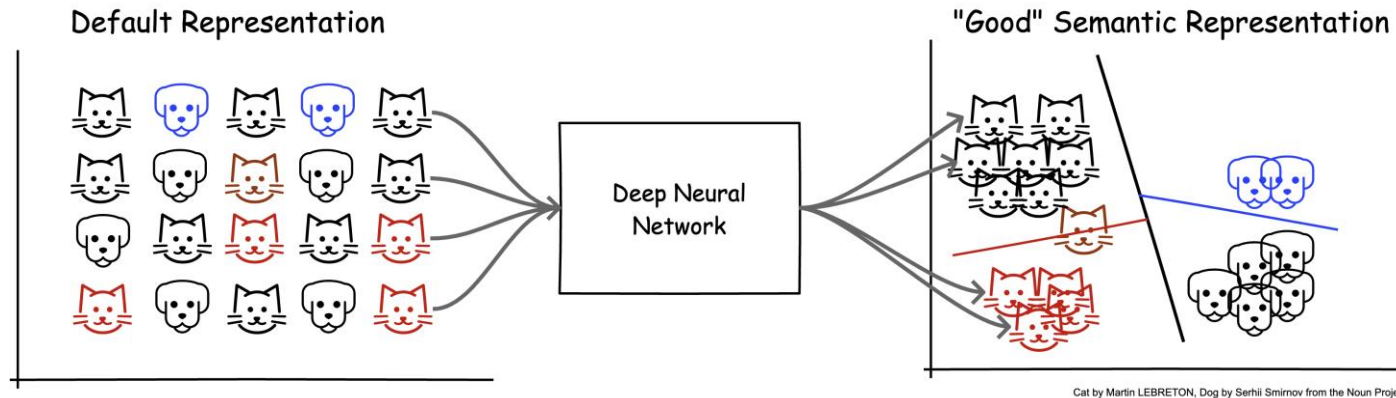
Overview

- Causality
 - 인과관계를 이용해 모델을 설명, 구축하려는 관점이 있음
- Representation learning
 - 데이터를 가장 잘 표현하는 representation vector를 구하려고 함
 - Neural net 을 이용해서..
- 위 두 개념을 결합한 논문
 - Causal mechanism 을 이용해 모델 설계
 - Causality 관점에서 기존 contrastive learning 을 설명

Contents

- Representation learning
- Causal mechanism
 - Cause variable
 - Intervention
 - do operator
- ReLIC

Representation learning

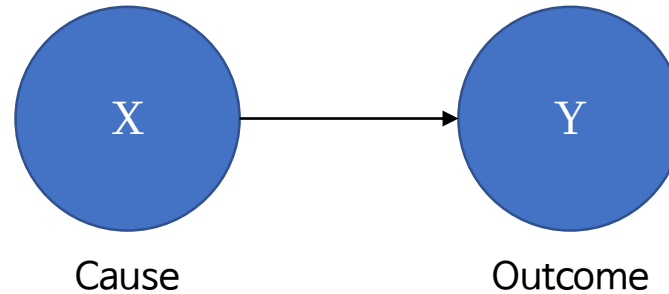


데이터를 가장 잘 표현하는 representation 을 찾고자 함

가장 잘 표현 : 이후 진행할 task 가 잘 수행될 수 있도록 하는 representation

Classification 문제라면 유사한 데이터끼리 뭉쳐 있는게 편할 것

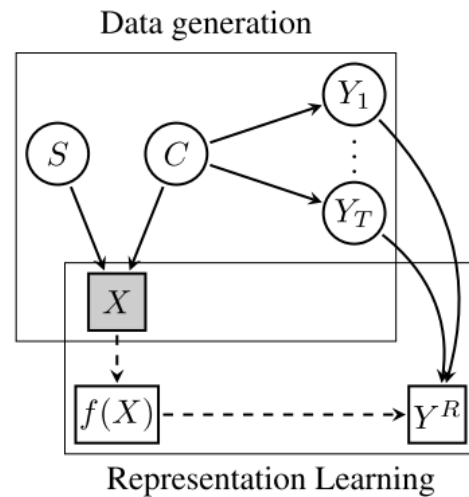
Causal Mechanism



X가 Y의 원인 이라면, X는 Y의 cause

만약, X를 맘대로 조작할 수 있어서 a라는 값으로 바꾸는 것 : $do(X=a)$

이렇게 do 연산을 해 X를 이것저것 바꿔가며 Y를 관찰하는 것 : intervention



S : Style

C : Content

X : Data

Y_t : Downstream task target

Y^R : Target

f : Model

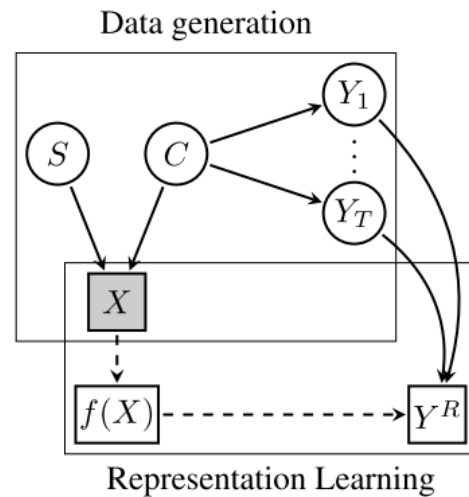
Style 과 content 로 인해 data 가 만들어 짐

Style 과 content 는 서로 독립

Content 만 downstream task (e.g. classification) 에 사용됨

Style 은 직접 바꿀 수 있음. Intervention 할 것.

(e.g. rotation, color distortion, noise ...)



S : Style

C : Content

X : Data

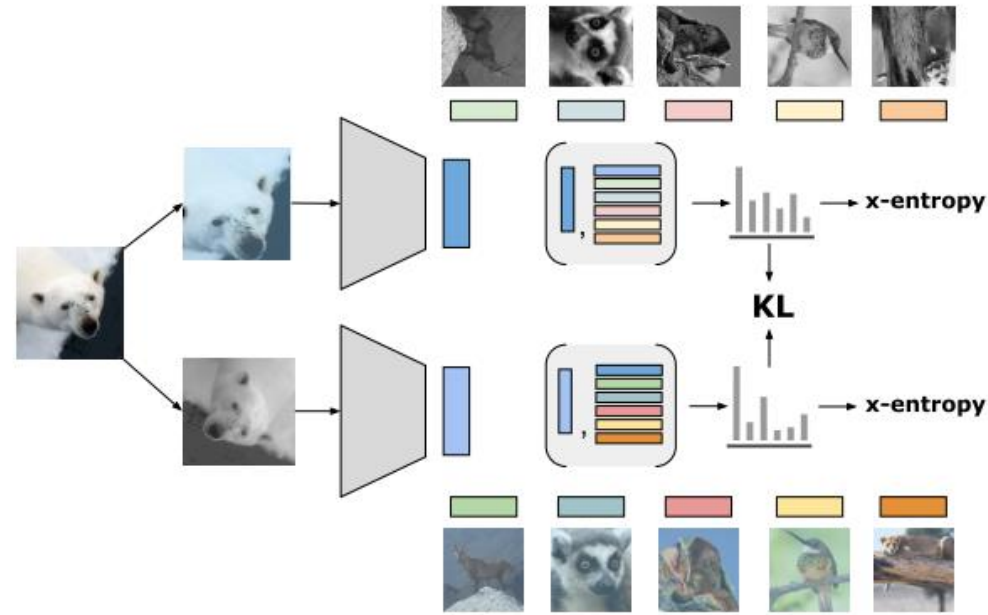
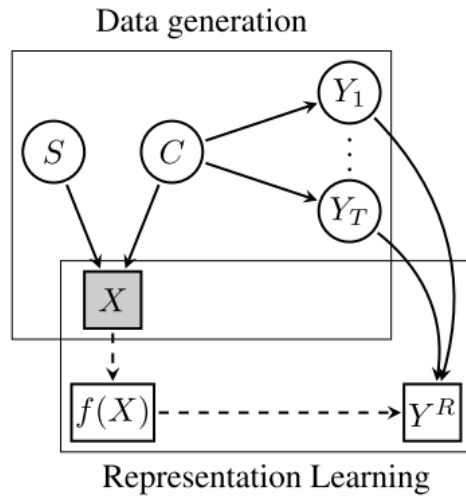
Y_t : Downstream task target

Y^R : Target

f : Model

Y_t 는 아직 모르니 downstream task 보다 더 어려운 proxy task 를 만들어 학습 (refinement)

$$Y^R \mid f(x_i) = i$$



Content 하나에 style 을 2개 적용

$$S \perp C \text{ 이므로 } P^{do(S=a)}(Y^R | C) = P^{do(S=b)}(Y^R | C)$$

KL + cross-entropy loss

패턴인식 논문발표

서울대학교 지능정보융합학과 음악오디오연구실
민대원

DEAR: Deep Reinforcement Learning for Online Advertising Impression in Recommender Systems

**Xiangyu Zhao¹, Changsheng Gu², Haoshenglun Zhang²
Xiwang Yang², Xiaobing Liu², Jiliang Tang¹, Hui Liu¹**

¹Michigan State University, ²Bytedance
{zhaoxi35,tangjili,liuhui7}@msu.edu

{guchangsheng,zhanghaoshenglun,yangxiwang,will.liu}@bytedance.com

AAAI 2021

1. Introduction

- ✓ DEAR: Deep Reinforcement Learning for Online Advertising Impression in Recommender Systems (Zhao *et al*, AAAI 2021)



1. Introduction

✓ DEAR: Deep Reinforcement Learning for Online Advertising Impression in Recommender Systems (Zhao *et al*, AAAI 2021)

- Tremendous interests in utilizing RL for online advertising in recommendation platforms
- However, the most RL-based algorithms focus on optimizing ad's revenue, ignoring the possible negative influences of ads on user experience of recommended items.
- For example, interpolating ads improperly or too frequently may decrease user experience, while interpolating fewer ads will reduce the advertising revenue.



1. Introduction

✓ DEAR: Deep Reinforcement Learning for Online Advertising Impression in Recommender Systems (Zhao *et al*, AAAI 2021)

- In this paper, a novel advertising strategy for the rec/ads trade-off is proposed.
- Specifically, it is the RL-based framework that can continuously update its advertising strategies and maximize reward in the long run.
- Given a recommendation list, the proposed Deep-Q-network(DQN) algorithm can determine three internally related tasks jointly;
 - 1) Whether to interpolate an ad of not in the recommendation list, and if yes,
 - 2) The optimal ad and,
 - 3) The optimal location to interpolate.

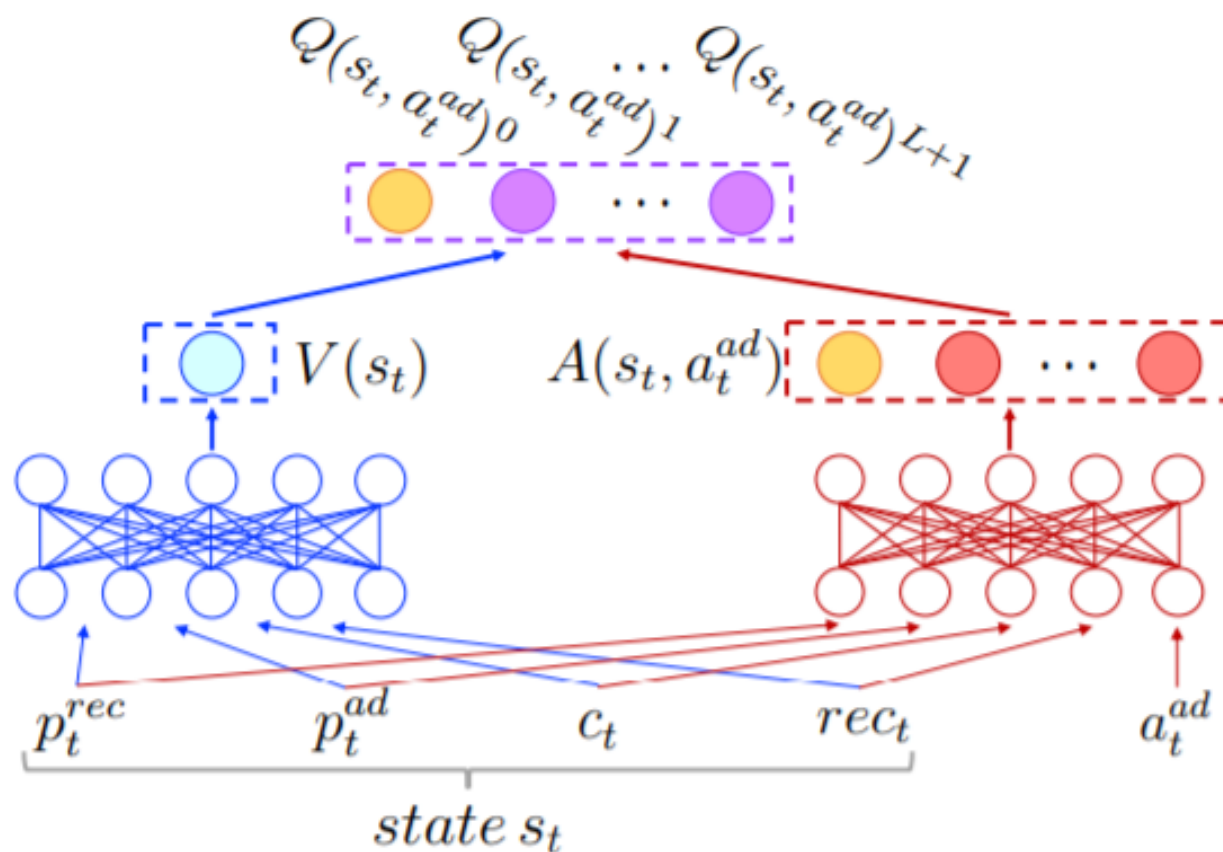


2. Problem Definition

✓ Problem Statement

- The advertising problem within a rec-list is considered as a Markov Decision Process(MDP), consisting of a tuple of five elements; (S, A, P, R, γ)
 - 1) State space S : A State $s_t \in S$ is defined as a user's browsing history before time t and the information of current request at time t .
 - 2) Action space A : The action $a_t = (a_t^{ad}, a_t^{loc}) \in A$ of AA is determine to three internally related tasks.
 - a_t^{ad} -> Chooses a specific ad.
 - a_t^{loc} -> Interpolates the chosen ad into the optimal location.
 - 3) Reward R : The reward $r(s_t, a_t)$ is two fold; (i) The income of an and that depends on the quality of the ad, and (ii) the influence of an ad on the user experience.
- Given the historical MDP; (S, A, P, R, γ) , the goal is to find an advertising policy $\pi: S \rightarrow A$, which can maximize the cumulative reward from users.

3. The Proposed Network



3. The Proposed Network

✓ The Processing of State and Action Features

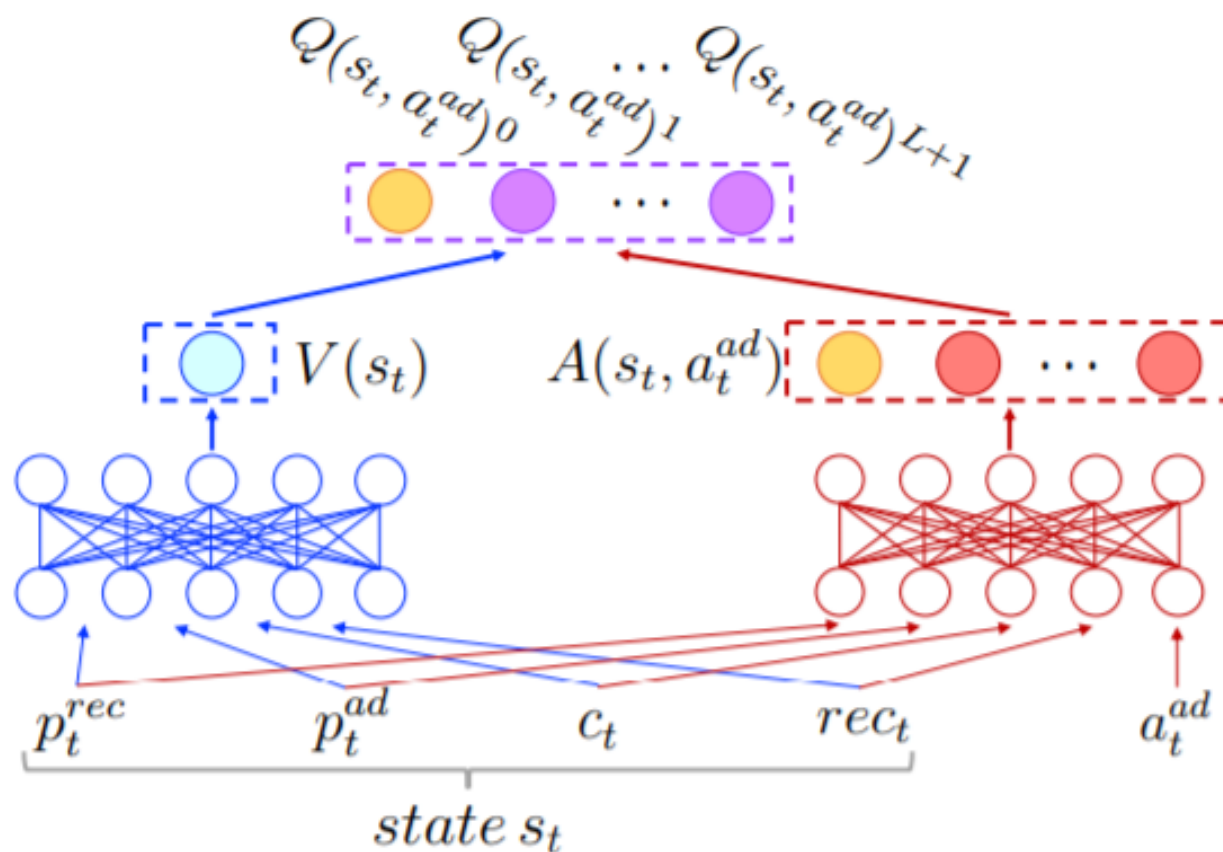
- $s_t = \text{concat}(p_t^{\text{rec}}, p_t^{\text{ad}}, c_t, \text{rec}_t)$
 - p_t^{rec} (or p_t^{ad})-> The final hidden state of GRU, which is the representation of user's dynamic preference of recommendations (or ads).
 - c_t -> The contextual information feature of current user request. It consists of information such as OS, app version and feed type (swiping up/down the screen).
 - rec_t -> The transformed low-dimensional dense vector of concatenated features of L recommended items displayed at time t , which is expressed as $\text{rec}_t = \tanh(W_{\text{rec}} \text{concat}(\text{rec}_1, \dots, \text{rec}_L) + b_{\text{rec}})$.
- For the action $a_t = (a_t^{\text{ad}}, a_t^{\text{loc}}) \in A$, a_t^{ad} is the feature of a candidate ad, and $a_t^{\text{loc}} \in R^{L+1}$ is the location to interpolate the selected ad (given the L items, there exist $L + 1$ possible locations).

3. The Proposed Framework

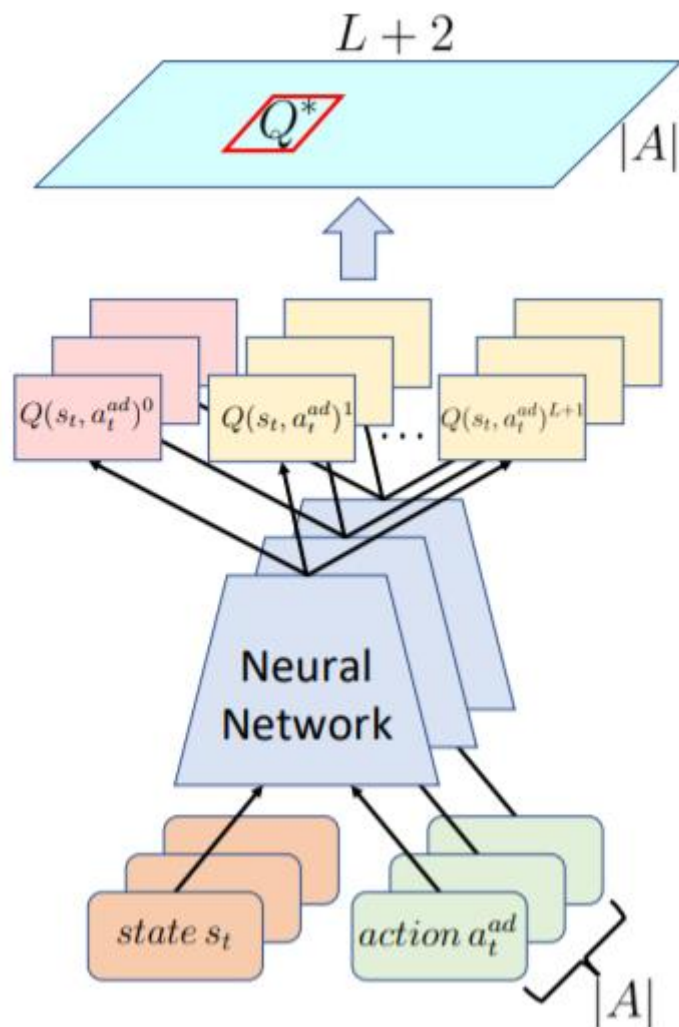
✓ The proposed DQN Architecture

- Firstly, to incorporate the sub-action (i) whether to interpolate an ad or not, into the DQN architecture, the option of not interpolating an ad is considered as a special *location* 0. The length of output layer is extended from $L + 1$ to $L + 2$, where $Q(s_t, a_t^{ad})^0$ corresponds to the Q-value of not incorporating an ad into current rec-list.
- Secondly, simultaneously tackle the sub-action (ii) where is the optimal and (iii) which ad is optimal. Those two sub-actions are integrated into one framework, estimating the Q-values of all possible locations a_t^{loc} for any given candidate ad a_t^{ad} .
- Can determine three internally related sub-actions at the same time !

3. The Proposed Network



3. The Proposed Network



3. The Proposed Framework

✓ The proposed DQN Architecture

- On the one hand, whether to interpolate an ad into current rec-list is mainly impacted by the state s_t (e.g, if a user has good experience for current rec-list, the advertising agent may prefer to interpolate an ad into the current rec-list).
- On the other hand, the reward for choosing an ad and location is closely related to all features (both current rec-list and the ads).
- According to these observations, the Q-function is divided into the value function $V(s_t)$, which is determined by th state features, and the advantage function $A(s_t, a_t)$, which is determined by the features of both state and action.

3. The Proposed Network

Algorithm 1 Off-policy Training of DEAR Framework.

- 1: Initialize the capacity of replay buffer \mathcal{D}
 - 2: Initialize action-value function Q with random weights
 - 3: **for** session = 1, M **do**
 - 4: Initialize state s_0 from previous sessions
 - 5: **for** $t = 1, T$ **do**
 - 6: Observe state $s_t = \text{concat}(p_t^{rec}, p_t^{ad}, c_t, rec_t)$
 - 7: Execute action a_t following off-policy $b(s_t)$
 - 8: Calculate reward $r_t = r_t^{ad} + \alpha r_t^{ex}$ from offline log
 - 9: Update state s_t to s_{t+1}
 - 10: Store transition (s_t, a_t, r_t, s_{t+1}) into \mathcal{D}
 - 11: Sample mini-batch of transitions (s, a, r, s') from \mathcal{D}
 - 12: Set

$$y = \begin{cases} r & \text{terminal } s' \\ r + \gamma \max_{a'} Q(s', a'; \theta) & \text{non-terminal } s' \end{cases}$$
 - 13: Minimize $(y - Q(s, a; \theta))^2$ according to Eq.(6)
 - 14: **end for**
 - 15: **end for**
-

Thank you



The continuous Bernoulli:
fixing a pervasive error in variational autoencoders

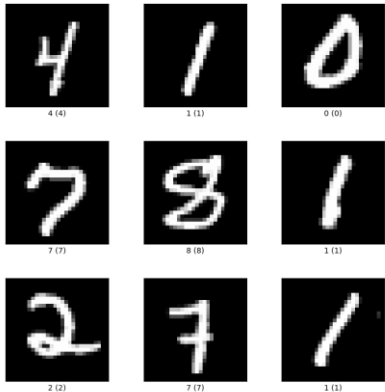
Gabriel Loaiza-Ganem and John P. Cunningham @ [NeurIPS 2020](#)



-
- Introduction
 - Method
 - Experiments
 - Conclusion

Introduction

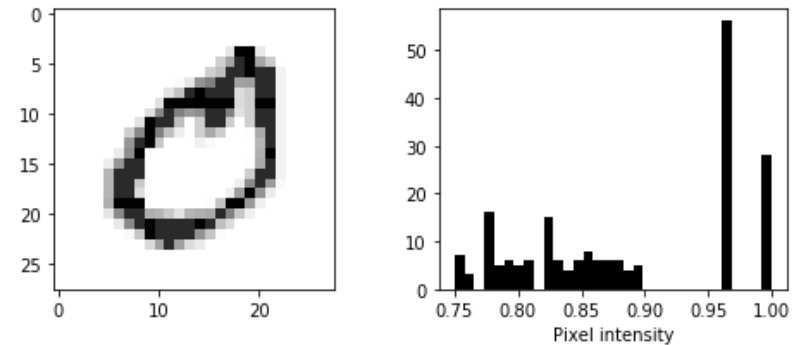
- Many works model the MNIST with neural network parameterizing Bernoulli likelihood
- MNIST dataset consists of images $x_n \in [0,1]^D, n = 1, \dots, N$ where N is the dataset size
- However, the support of Bernoulli distribution is $\{0,1\}$
 - VAE objective function built upon Bernoulli distribution is not a true probabilistic model
 - Modelling $x \in [0,1]$ as Bernoulli distribution **is equivalent to ignoring normalizing constant**



MNIST dataset samples

```
FeaturesDict({  
  'image': Image(shape=(28, 28, 1), dtype=tf.uint8),  
  'label': ClassLabel(shape=(), dtype=tf.int64, num_classes=10),  
})
```

In Tensorflow, MNIST is provided in integer type in $\{0, \dots, 255\}$, and we normalize it to $[0,1]$ float values.



From the pixel histogram we can see the MNIST do not follow pmf of Bernoulli

Introduction – cont'd

- How prior works address this problem?
 - Conduct data augmentation to binarize the MNIST data + negative binary cross entropy (non-probabilistic)
 - Another type of relaxation methods* have also been proposed
- Contribution of this work
 - Continuous Bernoulli (CB)-VAE outperforms Bernoulli (B)-VAE
 - It also outperforms Beta-distribution VAE and Gaussian VAE for $[0,1]$ ranged dataset
 - Provide decent lower bound that considers normalizing constant for *pdf*

*Maddison, C. J., Mnih, A., & Teh, Y. W. (2016). The concrete distribution: A continuous relaxation of discrete random variables. arXiv preprint arXiv:1611.00712.

Methods

- Evidence lower bound of variational autoencoder (VAE) for Bernoulli distribution

$$\mathcal{E}(\theta, \phi) = \sum_{n=1}^N E_{q_{\phi}(z_n|x_n)}[\log p_{\theta}(x_n|z_n)] - KL(q_{\phi}(z_n|x_n)||p_0(z_n)) \leq \log p_{\theta}((x_1, \dots, x_N)).$$

✓ $x_n \in [0,1]^D$ and we model $p_{\theta}(x_n|z_n) \rightarrow \mathcal{B}(\lambda_{\theta}(z_n))$, where $\lambda_{\theta}: \mathbb{R}^M \rightarrow \mathbb{R}^D$ is neural network θ

✓ $\mathcal{B}(\lambda_{\theta})$ is a product of D independent Bernoulli distribution (for all D pixels)

- **Problem:** Bernoulli distribution cannot model data $x_n \in (0,1)^D \rightarrow$ unnormalized in $[0,1]$

✓ $\tilde{p}(x|\lambda) = \lambda^x(1-\lambda)^x$, $\tilde{\cdot}$ denotes the distribution is unnormalized in $[0,1]$

✓ It is **equivalent to ignoring the normalizing constant**, which is crucial for probabilistic modelling*

- The continuous Bernoulli (CB), \mathcal{CB}

- It is a distribution having a support in $[0,1]$, parameterized by $\lambda \in (0,1)$.

$$X \sim \mathcal{CB}(\lambda) \iff p(x|\lambda) \propto \tilde{p}(x|\lambda) = \lambda^x(1-\lambda)^{1-x}.$$

Methods – cont'd

■ Properties of \mathcal{CB}

- *pdf*: normalized with $C(\lambda)$

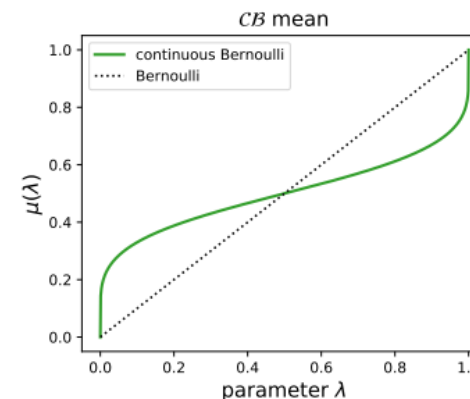
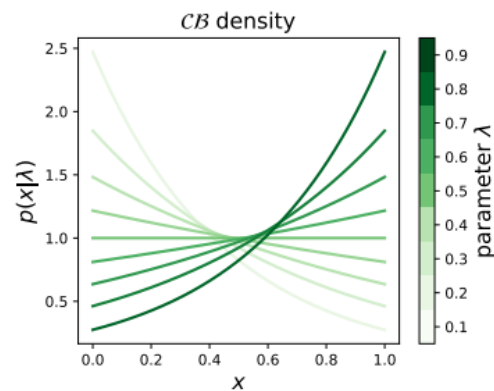
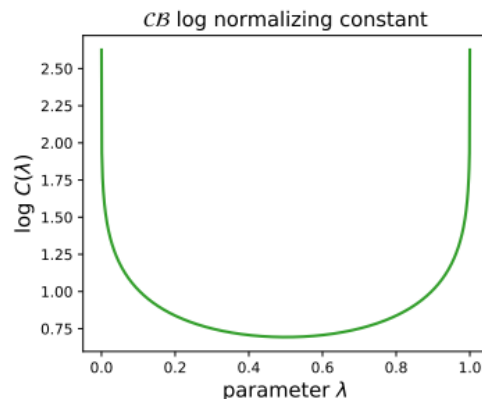
$$p(x|\lambda) = C(\lambda)\lambda^x(1-\lambda)^{1-x}, \text{ where } C(\lambda) = \begin{cases} \frac{2\tanh^{-1}(1-2\lambda)}{1-2\lambda} & \text{if } \lambda \neq 0.5 \\ 2 & \text{otherwise} \end{cases}$$

- *mean*

$$\mu(\lambda) := E[X] = \begin{cases} \frac{\lambda}{2\lambda-1} + \frac{1}{2\tanh^{-1}(1-2\lambda)} & \text{if } \lambda \neq 0.5 \\ 0.5 & \text{otherwise} \end{cases}$$

- *conjugate prior*

✓ $p(\lambda|\alpha, \beta, \nu) \propto \lambda^{\alpha-1}(1-\lambda)^{\beta-1}C(\lambda)^\nu$, where $\alpha, \beta > 0$ and $\nu \geq 0$



Methods – cont'd

■ Properties of \mathcal{CB}

- Forms an exponential family, closed form variance, CDF, and inverse CDF, and [entropy](#)

```
CLASS torch.distributions.continuous_bernoulli.ContinuousBernoulli(probs=None,  
    logits=None, lims=(0.499, 0.501), validate_args=None) [SOURCE]
```

```
cdf(value) [SOURCE]
```

```
PROPERTY mean
```

```
has_rsample = TRUE
```

```
log_prob(value) [SOURCE]
```

```
entropy() [SOURCE]
```

```
PROPERTY variance
```

```
icdf(value) [SOURCE]
```

- Bernoulli limit $\mathcal{CB}(\lambda) \xrightarrow{\lambda \rightarrow 0} \delta(0)$ and $\mathcal{CB}(\lambda) \xrightarrow{\lambda \rightarrow 1} \delta(1)$

✓ \mathcal{CB} becomes Bernoulli at each limit of the support

- Maximum likelihood

✓ From observed variables $x_1, \dots, x_N \sim_{\text{iid}} \mathcal{CB}(\lambda)$, MLE $\hat{\lambda}$ of λ is $\mu(\hat{\lambda}) = \frac{1}{N} \sum_n x_n$

- Thus, we can formulate decent ELBO for VAE on the \mathcal{CB}

$$\mathcal{E}(p, \theta, \phi) = \sum_{n=1}^N -KL(q_\phi || p_0) + E_{q_\phi} \left[\sum_{d=1}^D x_{n,d} \log \lambda_{\theta,d}(z_n) + (1 - x_{n,d}) \log(1 - \lambda_{\theta,d}(z_n)) + \log C(\lambda_{\theta,d}(z_n)) \right]$$

Experiments

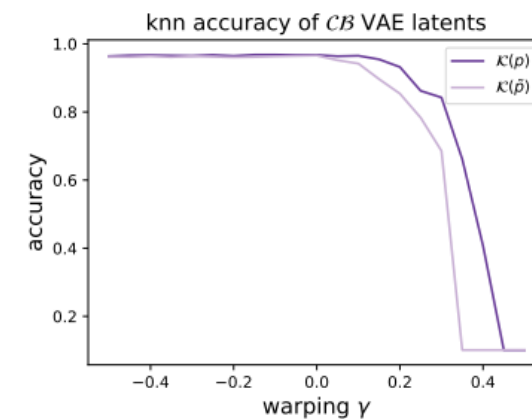
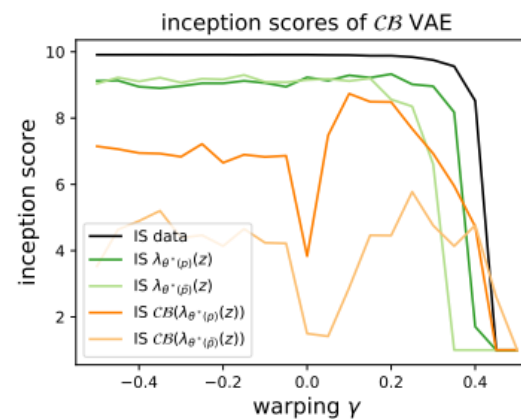
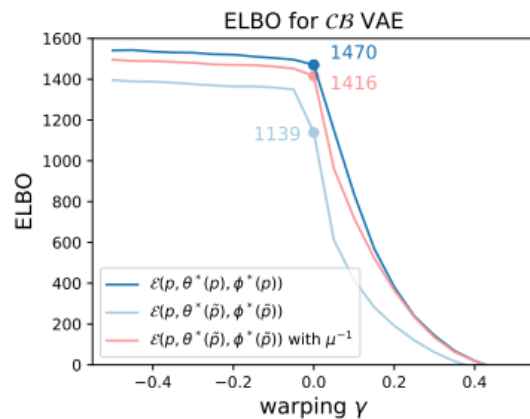
- Experimental results

- Qualitative reconstruction from VAEs

	data	6	0	5	8	7	6	2	0	8	3
Continuous Bernoulli	\mathcal{CB} VAE	5	3	6	2	7	8	9	1	0	9
Bernoulli	\mathcal{B} VAE	2	9	8	0	0	6	1	4	2	7
Gaussian	\mathcal{N} VAE	2	8	6	0	9	1	8	4	5	3

- Quantitative results γ denotes degree of warping the MNIST data; γ : -0.5 (binary), 0.0 (original), and > 0 (blurry)

✓ \mathcal{CB} shows higher ELBO, close Inception score to GT, and higher knn accuracy compared to \mathcal{B}



Conclusion

- Proposed a novel probability distribution with $[0,1]$ support
- Continuous Bernoulli resolves the normalization error of Bernoulli
- Continuous Bernoulli supports
 - Re parameterization
 - Likelihood evaluation
 - sampling
- VAE with continuous Bernoulli outperforms baselines (Bernoulli and Gaussian)

Thank you for your attention!



[CVPR'21]

Fast Bayesian Uncertainty Estimation and Reduction of Batch Normalized Single Image Super-Resolution Network

Aupendu Kar , Parbir Kumar Biswas

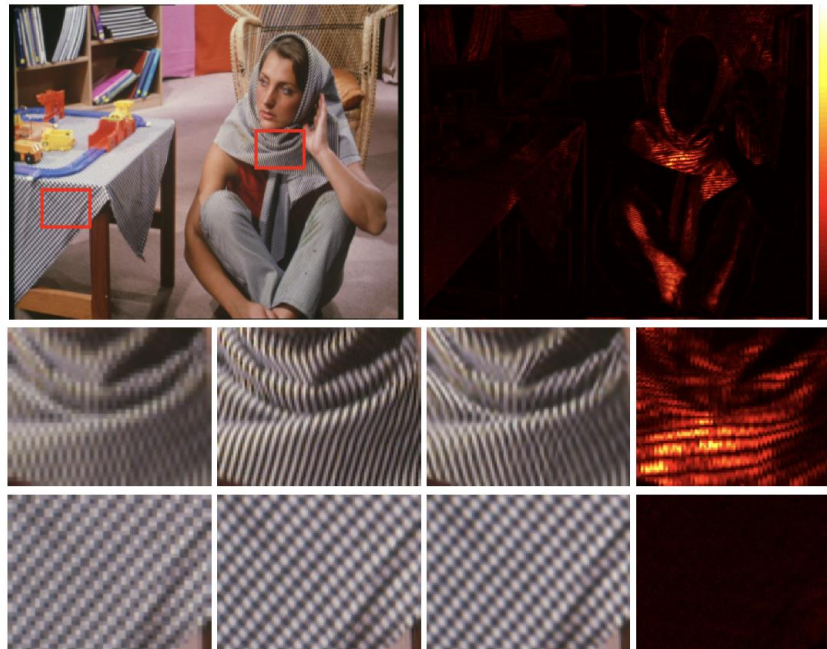
Indian Institute of Technology Kharagpur, WB, India
mailto:aupendu@gmail.com, pkb.iitkgp.ac.in

2016-12500

박기범

Introduction

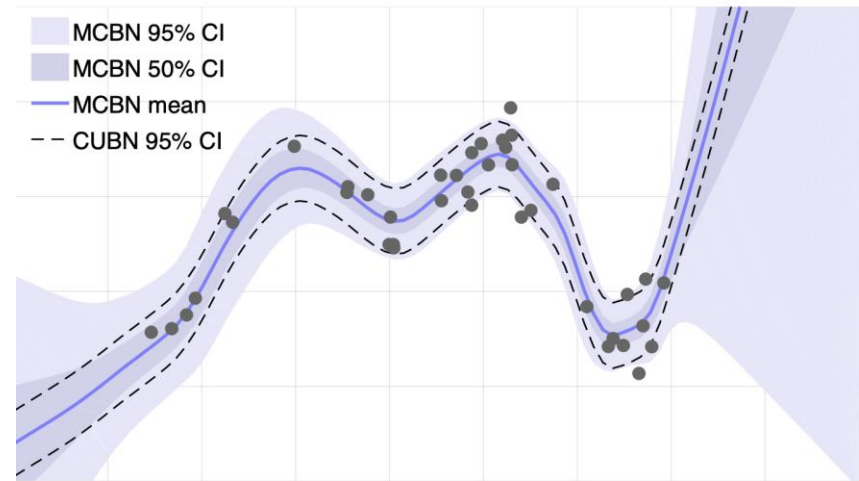
- Super Resolution DL models need more transparency to decide whether to trust the output
- Bayesian approaches can provide posterior distribution
- Author proposed faster Monte Carlo Batch normalization algorithm for SR provide the uncertainty
- Proposed algorithm to reduce uncertainty and Improve the result



Background | MCBN

- Batch normalization process
 1. Normalize $\hat{x}_i = \frac{x_i - \mu_B}{\sigma_B}$
 2. Scale & Shift $y_i = \gamma \hat{x}_i + \beta$Learnable = $\{W, \gamma, \beta, \mu_B, \sigma_B\}$
- Use $\{\mu_B, \sigma_B\}$ as stochastic parameter
And pick random sampled $\{\mu_B, \sigma_B\}$
from batch to get y
- Use $E[y]$ for result
 $Cov[y]$ for uncertainty
- We can get not only value but also
Uncertainty from model

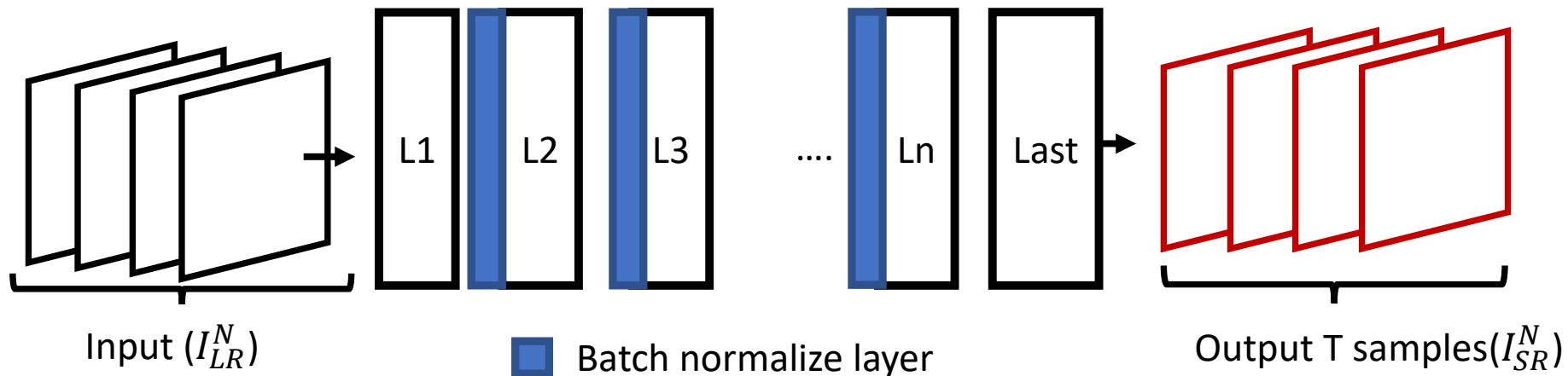
- But in Single Image Super Resolution problem making batches of test image is
too expensive



Algorithm and architecture |

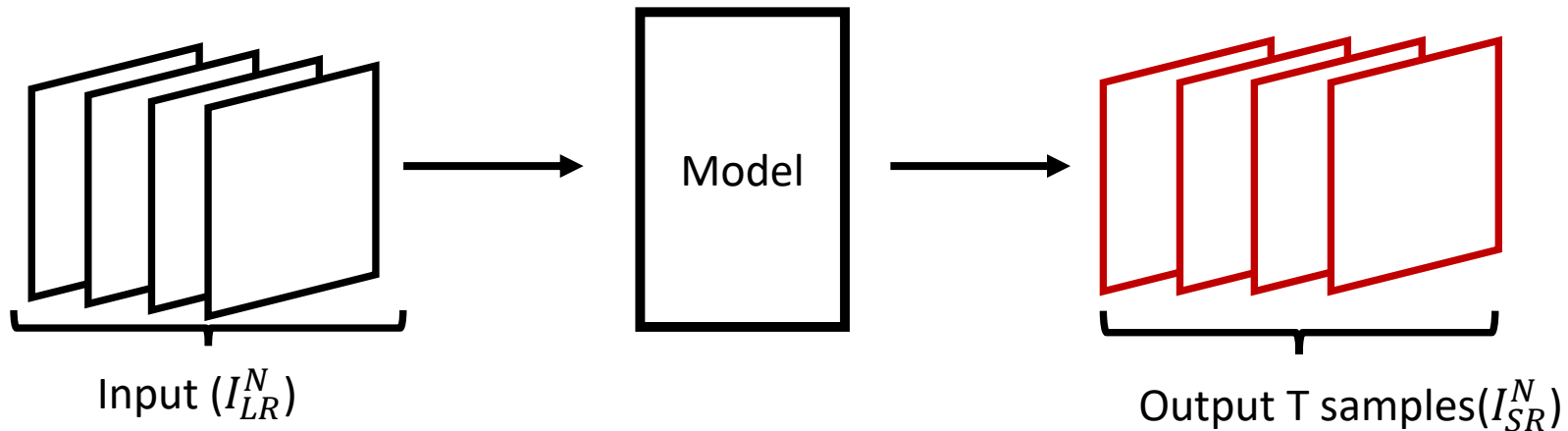
- Insert Batch normalization in every layer to existing SR model
- Use BN layer to estimate uncertainty
- Calculate mean to get output and Std to get uncertainty
- Concat input image as input and Instead of calculate $\{\mu_B, \sigma_B\}$ distribution from test batch we use $\{\mu_B, \sigma_B\}$ for faster calculation

$$I_{SR}^N = F_W(I_{LR}^N, \hat{w}_L^N);$$
$$\hat{I}_{SR} = \text{mean}(I_{SR}^N);$$
$$\hat{\sigma} = \text{std}(I_{SR}^N);$$

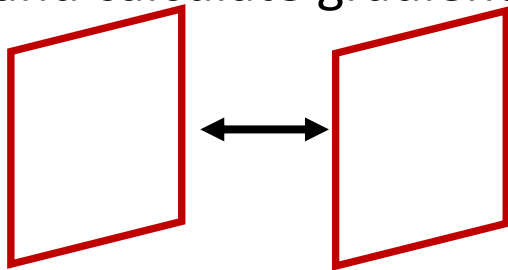


Uncertainty reduction |

- Sample T output with model



Random sample two output to calculate uncertainty T times and calculate gradient direction

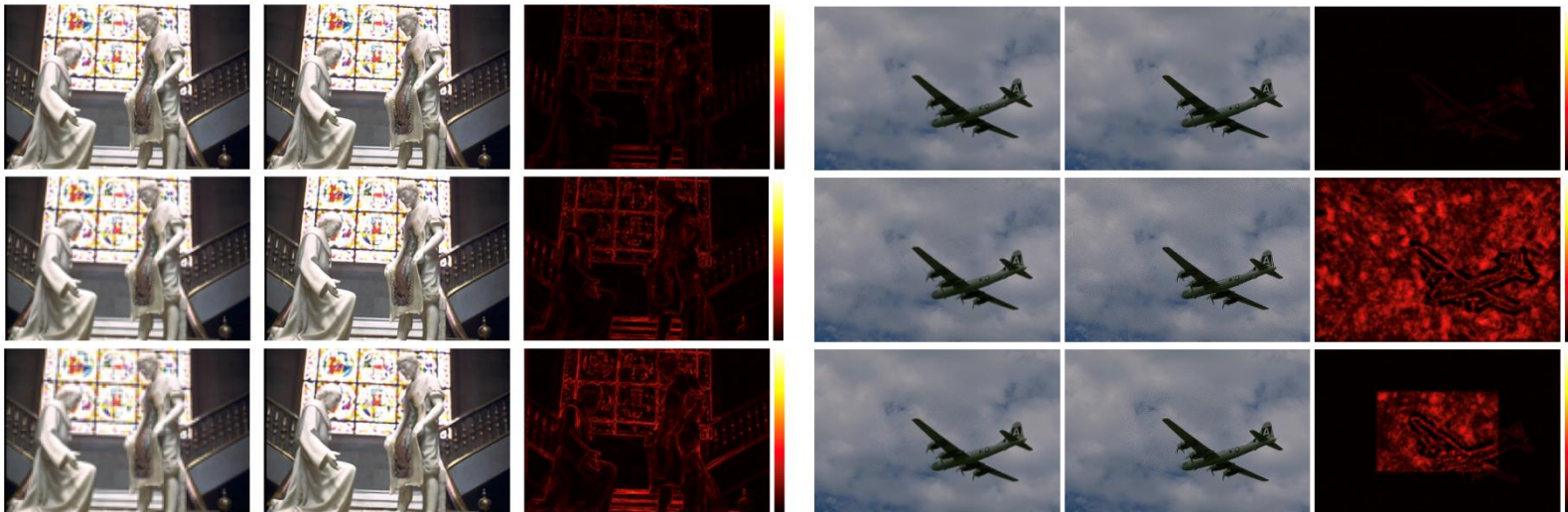


- Decide perturbate level β to find minimum uncertainty

```
for total updates on  $I_{LR}$  do
  Gradient directions for  $I_{LR}$ :
  for total iterations, T do
    Random Select  $t_1$  and  $t_2$ ;
     $I_{SR}^{t_1} = F_W(I_{LR}, \hat{w}_L^{t_1})$ ;
     $I_{SR}^{t_2} = F_W(I_{LR}, \hat{w}_L^{t_2})$ ;
     $\mathcal{L} = \frac{1}{CWH} \sum (I_{SR}^{t_1} - I_{SR}^{t_2})^2$ ;
     $\mathfrak{N} = \mathfrak{N} + \text{sign}(\nabla \mathcal{L})$ 
  end
end
for different levels of perturbation,  $\beta$  do
   $I'_{LR} = I_{LR} - \beta \cdot \frac{\mathfrak{N}}{T}$ ;
  Calculate Uncertainty for  $I'_{LR}$ ,  $U(\beta)$ ;
  if  $U(\beta) > U(\beta - 1)$  then
     $I_{LR} = I_{LR} - (\beta - 1) \cdot \frac{\mathfrak{N}}{T}$ ;
    break;
```

Result | Example

- Increasing SR scale increase uncertainty
- Noise unseeable in LR can make high uncertainty in SR



(a) LR Image

(b) SR Image

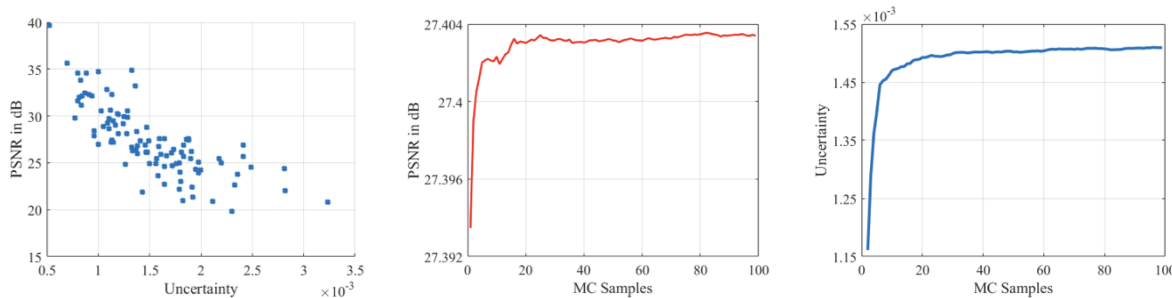
(c) Uncertainty

(a) LR Image

(b) SR Image

(c) Uncertainty Map

- PSNR decrease with increasing uncertainty



Result | measurement

- Noise made in LR can be reduced by uncertainty reduction



(a) HR

(b) LR

(c) Adversarial LR

(d) SR (No Def.)

(e) SR (After Def.)

(f) UN (No Def.)

(g) UN (After Def.)

Type	Scale Factor ($\times 2/ \times 3/ \times 4$)				
	Level of Attack	PSNR		Uncertainty	
		No Defense	After Defense	No Defense	After Defense
No Attack	-	32.02/ 28.95/ 27.43	31.94/ 28.87/ 27.32	0.0011/ 0.0014/ 0.0015	0.0009/ 0.0011/ 0.0013
Attack	1	31.16/ 28.36/ 27.03	31.55/ 28.63/ 27.16	0.0014/ 0.0017/ 0.0017	0.0010/ 0.0013/ 0.0013
	2	29.54/ 27.27/ 26.25	30.86/ 28.25/ 26.89	0.0019/ 0.0022/ 0.0021	0.0013/ 0.0014/ 0.0015
	4	26.30/ 25.03/ 24.63	29.73/ 27.67/ 26.43	0.0034/ 0.0035/ 0.0030	0.0018/ 0.0018/ 0.0018
	8	21.97/ 21.78/ 22.02	27.78/ 26.60/ 25.59	0.0058/ 0.0055/ 0.0047	0.0026/ 0.0024/ 0.0023
	16	18.04/ 18.37/ 18.82	24.78/ 24.64/ 23.94	0.0086/ 0.0084/ 0.0072	0.0040/ 0.0037/ 0.0035

Thank you

BAYES-TREX: a Bayesian Sampling Approach to Model Transparency by Example

Serena Booth, Yilun Zhou, Ankit Shah, Julie Shah

AAAI 2021

발표자: 박상하

Seoul National University

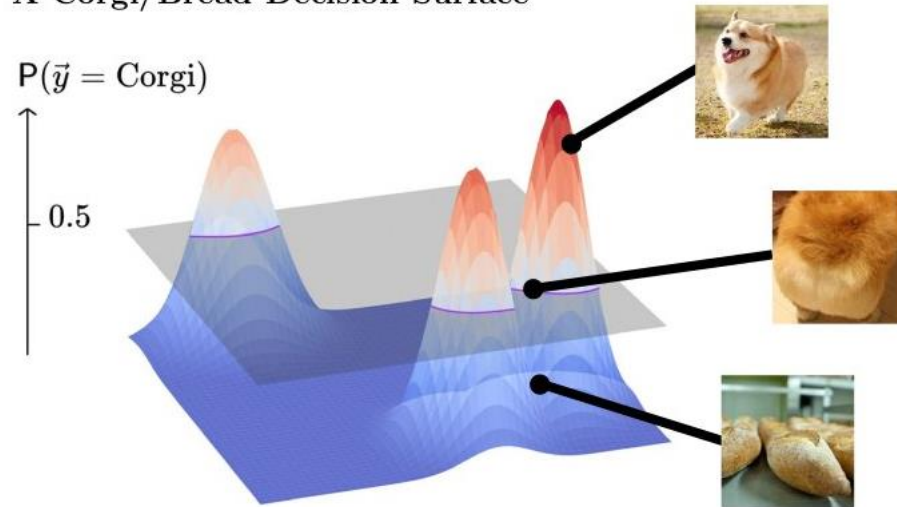
Data Science & Artificial Intelligence Laboratory

2021.11.20

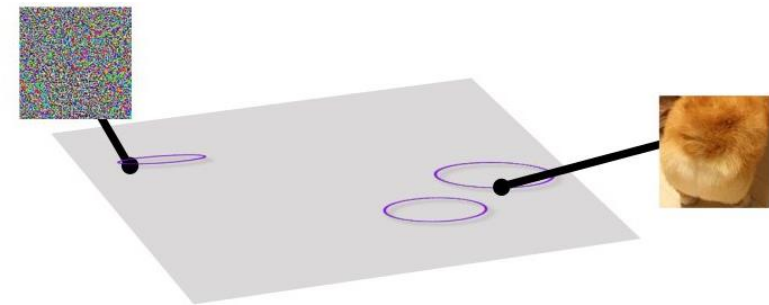
Motivation

- Transparency by example
- Objective: Build a *holistic* understanding of a classifier.

A Corgi/Bread Decision Surface



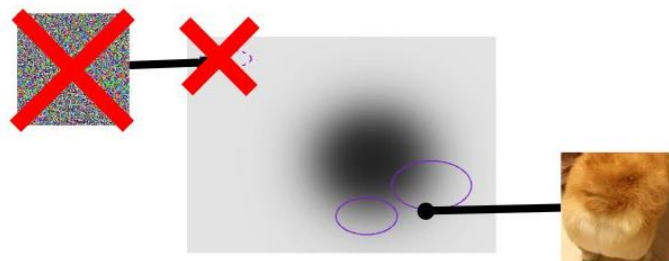
- **Bayes-TrEx** search for level set examples which elicit a target prediction to help us gain insight into the classifier.



This slice corresponds to $P(\text{Corgi}) = 0.5$ level set.

Method

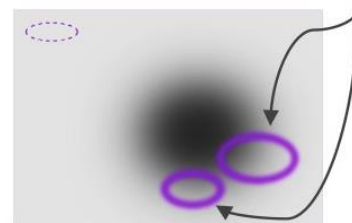
- Bayes-TrEx want to find an example \mathbf{x} which is natural and plausible under the data, and for which the classifier $f(\mathbf{x})$ has confidence \mathbf{p}



Want to sample from: $p(\mathbf{x}|f(\mathbf{x}) = \mathbf{p}) \propto p(\mathbf{x})(f(\mathbf{x}) = \mathbf{p}|\mathbf{x})$

- Two problems** in applying MCMC methods:
 - $\{\mathbf{x}: f(\mathbf{x}) = \mathbf{p}\}$ has small or even zero measure.
 - \mathbf{x} too high dimensional.

To solve **problem 1**, Bayes-TrEx relax the formulation by widening the level set.

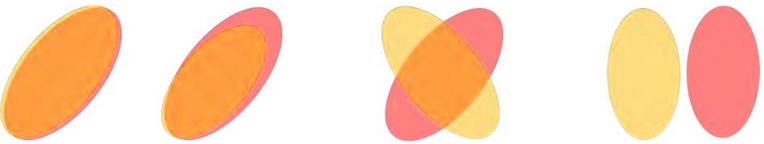


Introduce a random vector:
 $\mathbf{u}|\mathbf{x} \sim \mathcal{N}(f(\mathbf{x}), \sigma^2)$

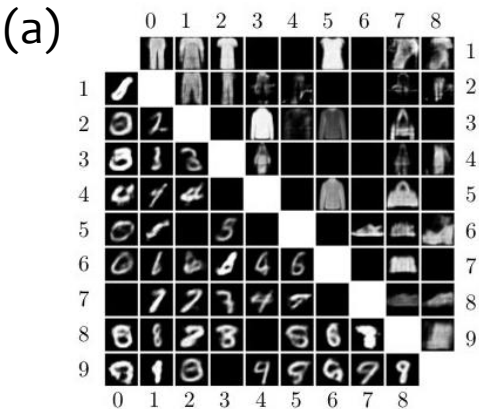
And sample from the new posterior:
 $p(\mathbf{x}|\mathbf{u} = \mathbf{u}^*) \propto p(\mathbf{x})p(\mathbf{u} = \mathbf{u}^*|\mathbf{x})$
 $\mathbf{u}^* = \mathbf{p}$

To solve **problem 2**, Bayes-TrEx use a generative model to represent \mathbf{x} and sample from its parameter space, instead.

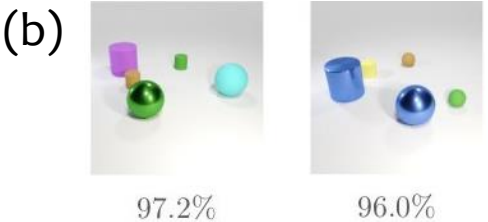
Applications, Evaluation, and Results



(a) $P_D = P_C$ (b) $P_D \subset P_C$ (c) $P_D \cap P_C \neq P_D \neq P_C$ (d) $P_D \cap P_C = \emptyset$



Bayes-TrEx lets us assess **class boundaries** by finding examples where the classifier has 50/50 confidence between two classes.



Classifier: "Contains 1 Cube"

Bayes-TrEx lets us find **high confidence failures**, which are more likely to be missed in assessing models.



Classifier: "Contains 5 Cube"

Bayes-TrEx lets us assess model responses to **novel classes**, like these Corgis.



Further assessing with explanation method



SIMULATION-DRIVEN STRUCTURE
DESIGN LABORATORY

Disentangling Label Distribution for Long-tailed Visual Recognition

Youngkyu Hong Seungu Han Kwanghee Choi Seokjun Seo Beomsu Kim Buru Chang
Hyperconnect

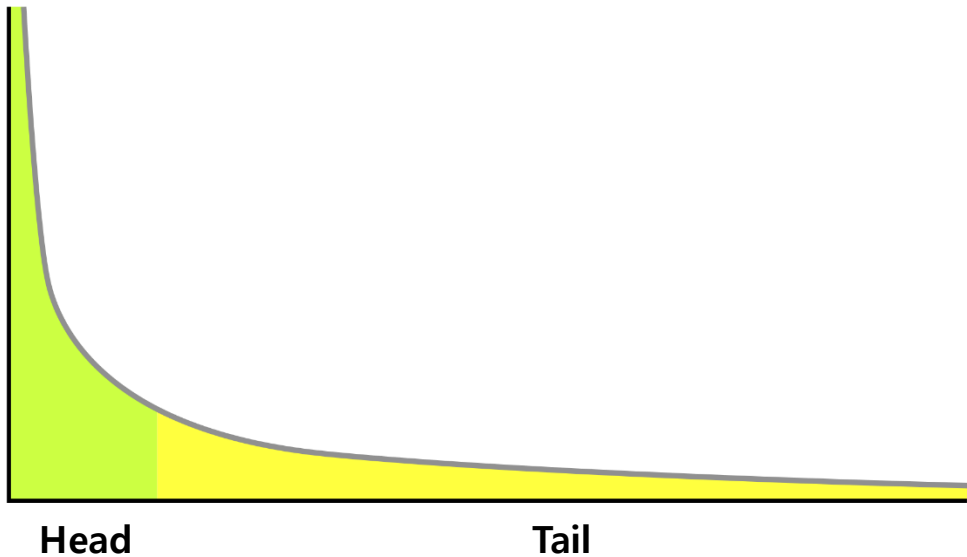
서울대학교 기계공학부
2021-26800
박선재



- Problem definition

Long Tail Problem in CS

Head에 해당하는 데이터는 많지만 Tail에 해당하는 데이터의 개수가 부족한 경우
Class간 데이터 분포가 균일한 경우보다 성능이 저하되는 문제



Long tail Dataset

Entanglement

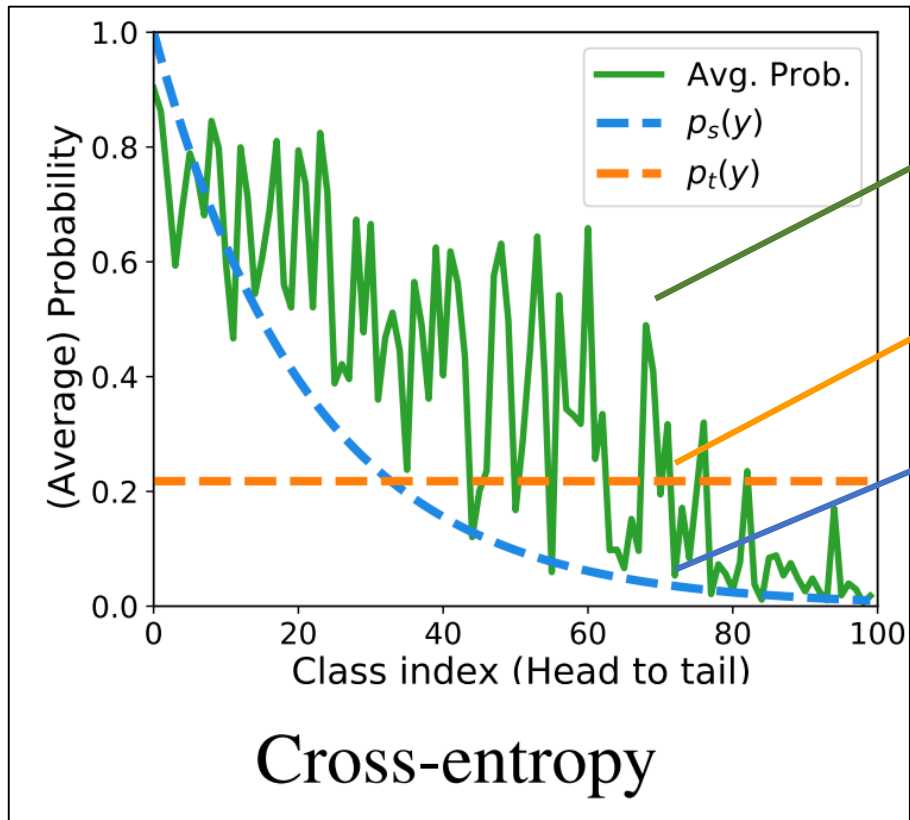
$P_S(y)$
학습 데이터 분포



모델 예측



- Problem definition



모델 예측 확률의 평균이 데이터의 개수가 감소함에 따라 같이 감소함

테스트 데이터는 클래스 당 균일한 분포를 가지고 있음

학습 데이터는 Class의 index가 100에 가까워질수록 클래스 당 데이터의 개수가 감소함

In $P_s(y) \neq P_t(y)$ Problem,

∴ Label distribution DisEntangle (LADE) Loss 를 이용하여 학습하면 성능을 향상 시킬 수 있다.



- Method

1. Post-Compensated Softmax

- 추론 단계에서 Soft max의 출력에 해당 클래스의 학습 데이터와 테스트 데이터 간 비율을 곱하여 추론함

$$P(y|x; \theta) = \frac{e^{f_{\theta}(x)[y]}}{\sum_c e^{f_{\theta}(x)[c]}}$$

Logit : $f_{\theta}^{PC}(x)[y] = f_{\theta}(x)[y] - \log p_s(y) + \log p_t(y)$

$$P_t(y|x; \theta) = \frac{\frac{p_t(y)}{p_s(y)} \cdot e^{f_{\theta}(x)[y]}}{\sum_c \frac{p_t(c)}{p_s(c)} \cdot e^{f_{\theta}(x)[c]}} = \frac{e^{(f_{\theta}(x)[y] - \log p_s(y) + \log p_t(y))}}{\sum_c e^{(f_{\theta}(x)[c] - \log p_s(c) + \log p_t(c))}} = \frac{e^{f_{\theta}^{PC}(x)[y]}}{\sum_c e^{f_{\theta}^{PC}(x)[c]}}$$



- Method

2. LADER: Label distribution DisEntangling Regularizer

$$p_s(y|x) = \frac{p_s(x|y)}{p_s(x)} \underbrace{p_s(y)}_{\text{Entangled!}}$$

$$\rightarrow p_s(y|x) = \frac{p_s(x|y)}{p_s(x)}$$

$$\rightarrow p_s(y|x) = \frac{p_s(x|y)}{p_s(x)} \rightarrow \frac{p_u(x|y)}{p_u(x)}$$

Replacing $p_s(y)$ with $p_u(y)$

$$f_\theta(x)[y] = \log \frac{p_u(x|y)}{p_u(x)} \leftarrow \text{DV representation 이용!}$$



- Donsker varadhan representation

$$D_{KL}(P||Q) \geq \sup_{T \in \mathcal{F}} E_P[T] - \log E_Q[e^T]$$

$$\log \frac{dP}{dQ} = \operatorname{argmax}_{T: \Omega \rightarrow \mathbb{R}} (E_P[T] - \log(E_Q[e^T]))$$

$$\log \frac{dP}{dQ} = \operatorname{argmax}_{T: \Omega \rightarrow \mathbb{R}} (E_P[T] - \log(E_Q[e^T]) - \lambda(\log(E_{x \sim p_u(x)}[e^T]))^2)$$

$$f_{\theta}(x)[y] = \log \frac{p_u(x|y)}{p_u(x)}$$



- Method

2. LADER: Label distribution DisEntangling Regularizer

$$f_{\theta}(x)[y] = \log \frac{p_u(x|y)}{p_u(x)}$$



각각의 항을 대응

$$\log \frac{dP}{dQ} = \underset{T: \Omega \rightarrow R}{\operatorname{argmax}} (E_p[T] - \log(E_Q[e^T]) - \lambda (\log(E_{x \sim p_u(x)}[e^T]))^2)$$

$$P \rightarrow p_u(x|y)$$

$$Q \rightarrow p_u(x)$$

$$T \rightarrow f_{\theta}(x)[y]$$



- Method

2. LADER: Label distribution DisEntangling Regularizer

$$\log \frac{p_u(x|y)}{p_u(x)} \geq \operatorname{argmax}_{f_\theta} E_{x \sim p_u(x|y)} [f_\theta(x)[y]] - \log(E_{x \sim p_u(x)} [e^{f_\theta(x)[y]}]) - \lambda (\log(E_{x \sim p_u(x)} [e^{f_\theta(x)[y]}]))^2$$

↓

$$\frac{1}{N_c} \sum_{i=1}^N 1_{y_i=c} \cdot f_\theta(x_i)[c]$$

↓ ↓

$$E_{(x,y) \sim p_s(x,y)} \left[\frac{p_u(y)}{p_s(y)} e^{f_\theta(x)[c]} \right] \approx \frac{1}{N} \sum_{i=1}^N \frac{p_u(y_i)}{p_s(y_i)} \cdot e^{f_\theta(x_i)[c]}$$

1. Monte Carlo approximation

2. Importance sampling



- Method

2. LADER: Label distribution DisEntangling Regularizer

$$L_{LADER_c} = -\frac{1}{N_c} \sum_{i=1}^N \mathbf{1}_{y_i=c} \cdot f_{\theta}(x_i)[c] - \log \left(\frac{1}{N} \sum_{i=1}^N \frac{p_u(y_i)}{p_s(y_i)} \cdot e^{f_{\theta}(x_i)[c]} \right) - \lambda \left(\log \left(\frac{1}{N} \sum_{i=1}^N \frac{p_u(y_i)}{p_s(y_i)} \cdot e^{f_{\theta}(x_i)[c]} \right) \right)^2$$

$$L_{LADER} = \sum_{c \in \mathcal{S}} \alpha_c \cdot L_{LADER_c}$$



- Method

2. LADER: Label distribution DisEntangling Regularizer

$$L_{LADE-CE}(f_{\theta}(x), y) = -\log(p_s(y|x; \theta)) = -\log\left(\frac{p_s(y) \cdot e^{f_{\theta}(x)[y]}}{\sum_c p_s(c) \cdot e^{f_{\theta}(x)[c]}}\right)$$

+

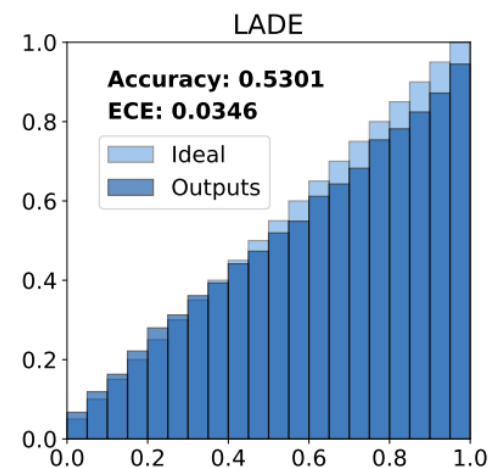
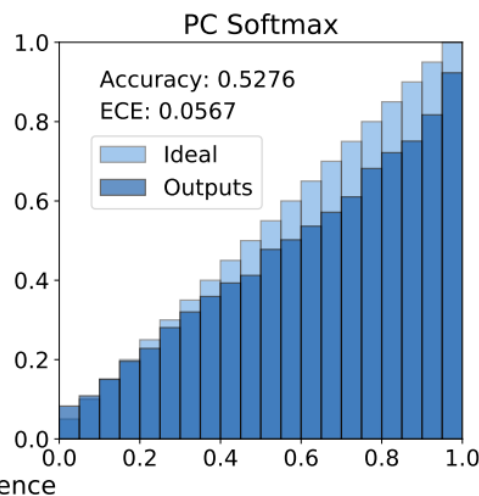
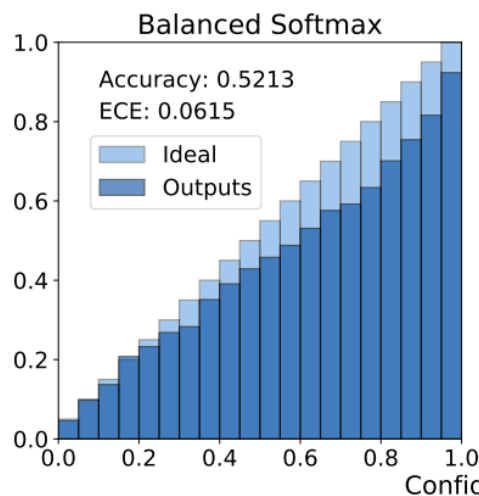
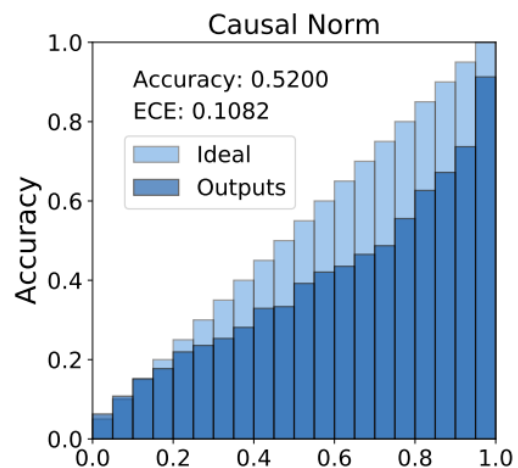
$$L_{LADER} = \sum_{c \in S} \alpha_c \cdot L_{LADER_c}$$

||

$$L_{LADE}(f_{\theta}(x), y) = L_{LADE-CE}(f_{\theta}(x), y) + \alpha \cdot L_{LADER}(f_{\theta}(x), y)$$



- Result



Dataset	CIFAR-100 LT		
	100	50	10
Focal Loss [†]	38.4	44.3	55.8
LDAM [†]	42.0	46.6	58.7
BBN [†]	42.6	47.0	59.1
Causal Norm [†]	44.1	50.3	59.6
Balanced Softmax	45.1	49.9	61.6
Softmax	41.0	45.5	59.0
PC Softmax	45.3	49.5	61.2
LADE	45.4	50.5	61.7

CIFAR-100LT

Method	Many	Medium	Few	All
<i>90 epochs</i>				
Focal Loss [§]	64.3	37.1	8.2	43.7
OLTR [§]	51.0	40.8	20.8	41.9
Decouple-cRT [§]	61.8	46.2	27.4	49.6
Decouple- τ -norm [§]	59.1	46.9	30.7	49.4
Decouple-LWS [§]	60.2	47.2	30.3	49.9
Causal Norm [§]	62.7	48.8	31.6	51.8
Balanced Softmax	62.2	48.8	29.8	51.4
Softmax	65.1	35.7	6.6	43.1
PC Softmax	60.4	46.7	23.8	48.9
LADE	62.3	49.3	31.2	51.9
<i>180 epochs</i>				
Causal Norm	65.2	47.7	29.8	52.0
Balanced Softmax	63.6	48.4	32.9	52.1
Softmax	68.1	41.9	14.4	48.2
PC Softmax	63.9	49.1	34.3	52.8
LADE	65.1	48.9	33.4	53.0

ImageNet-LT

Method	Top-1 Accuracy
CB-Focal [†]	61.1
LDAM [†]	64.6
LDAM+DRW [†]	68.0
Decouple- τ -norm [†]	69.3
Decouple-LWS [†]	69.5
BBN*	69.6
Causal Norm	63.9
Balanced Softmax	69.8
Softmax	65.0
PC Softmax	69.3
LADE	70.0

iNaturalist 2018



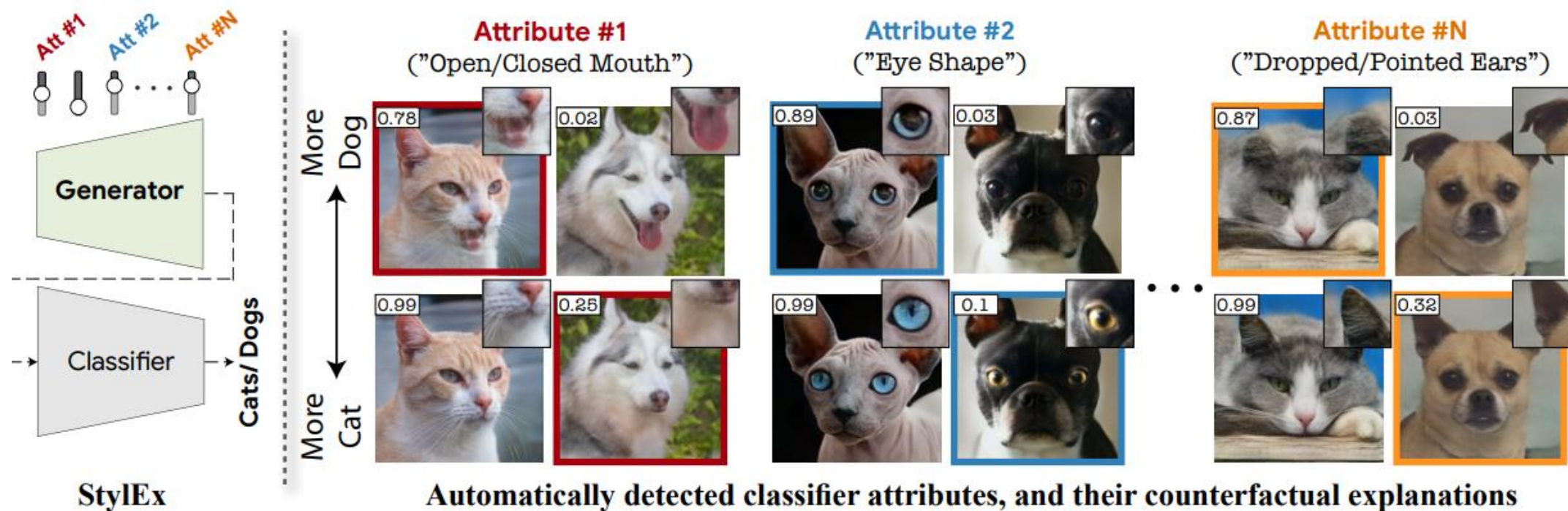
Explaining in Style: Training a GAN to explain a classifier in StyleSpace

ICCV 2021

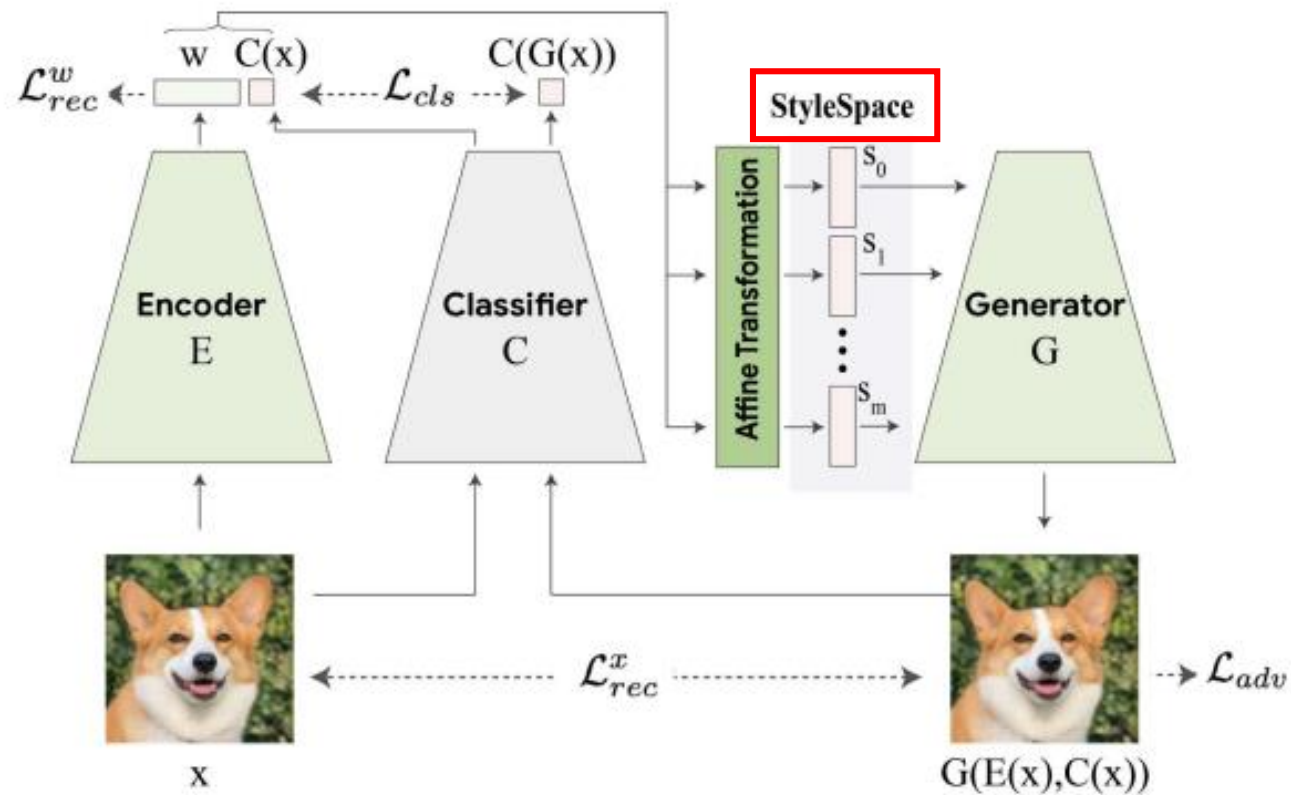
발표자 : 전기정보공학부 박선지

Goal

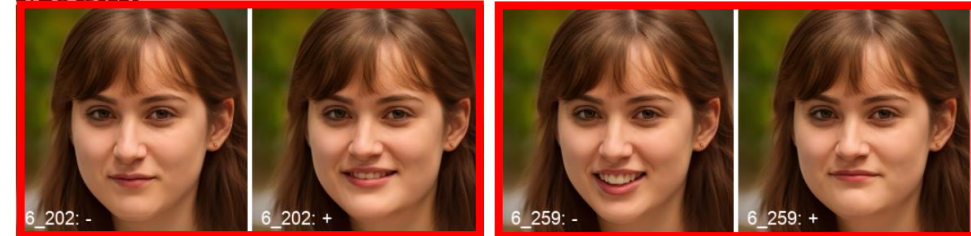
: Explain the classification of a given image by changing certain **attributes** in the image



StyleSpace



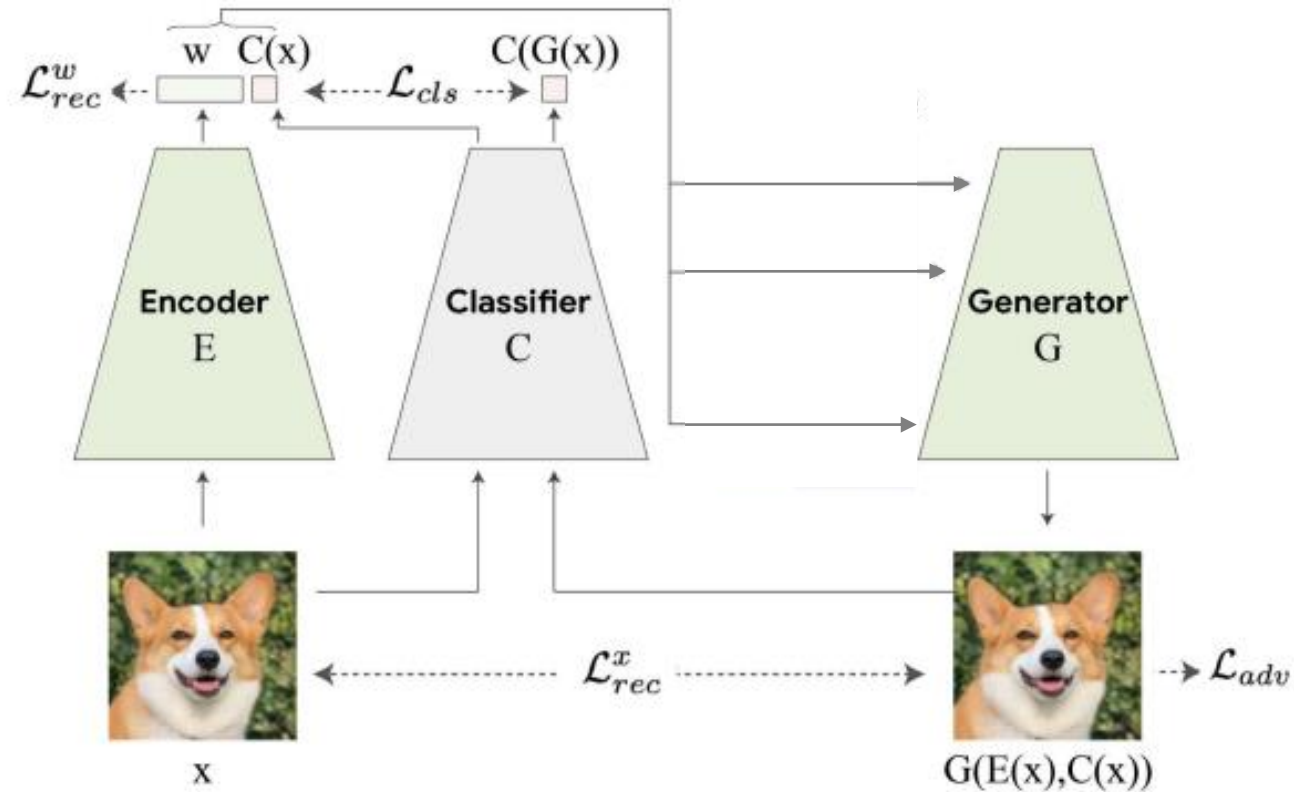
Mouth:



Attribute Finding

- To Find Cat-specific attributes (P_{cat} = Probability of being classified as a cat)
 1. For N Dog images, find attribute (s_1) that best maximizes P_{cat}
 2. Removes all images that changing s_1 had a large effect to P_{cat}
 3. Remove s_1 from StyleSpace set
 4. Do step 1,2,3 until s_M or there is no image left

StyleEx Architecture

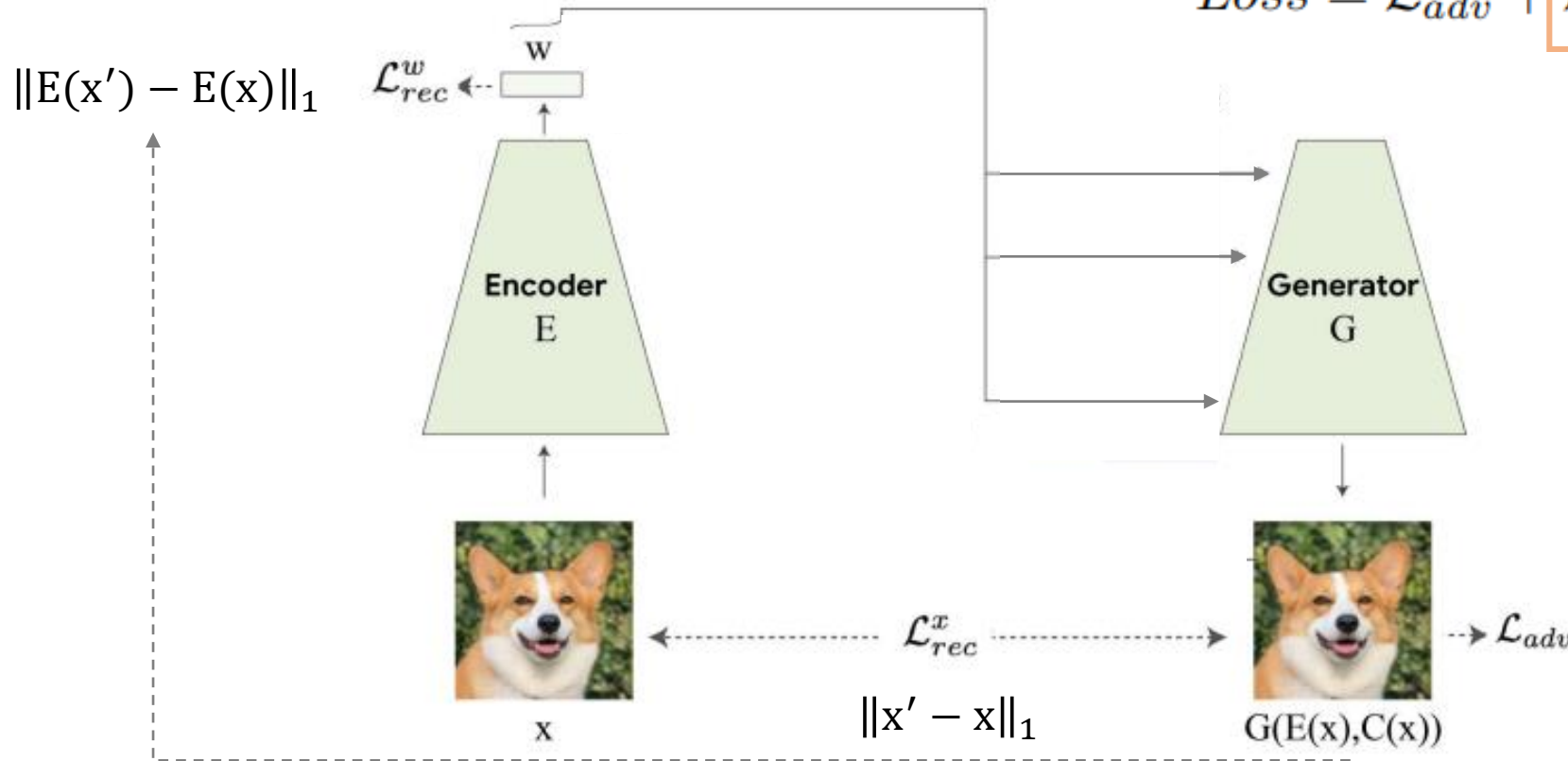


StyleEx Architecture (Same with StyleGAN2)

L_{reg} = PPL Regularization from styleGAN2

$$Loss = \mathcal{L}_{adv} + \mathcal{L}_{reg} + \mathcal{L}_{rec}$$

$$\begin{aligned} \mathcal{L}_{rec} &= \mathcal{L}_{rec}^x + L_{LPIPS} + \mathcal{L}_{rec}^w \\ &= \|x' - x\|_1 \\ &\quad + (\text{LPIPS distance of } x \text{ and } x') \\ &\quad + \|E(x') - E(x)\|_1 \end{aligned}$$

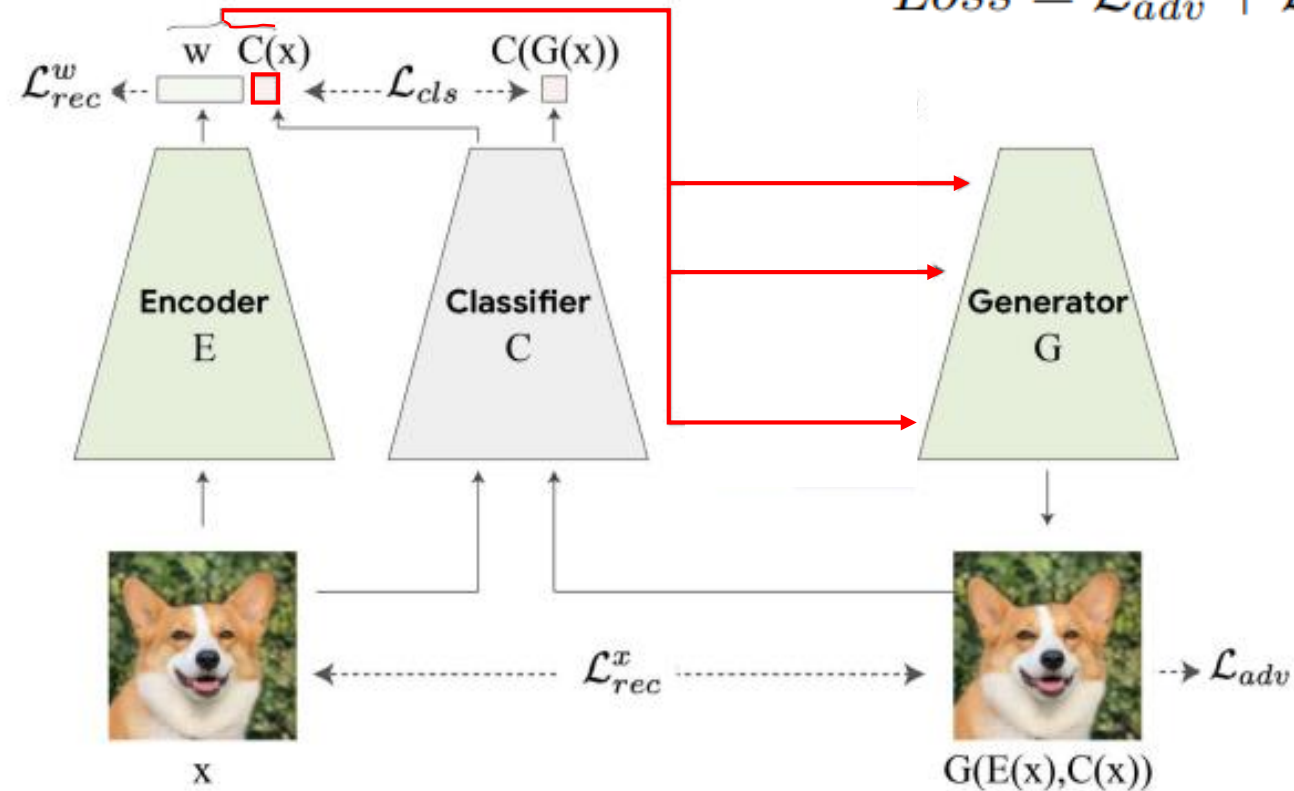


StyleEx Architecture

$$\mathcal{L}_{cls} = D_{KL} [C(x') || C(x)].$$

: KL-divergence between the classifier output on the **generated image**, and the **original input image**.
=> **Make them to be classified as same class**

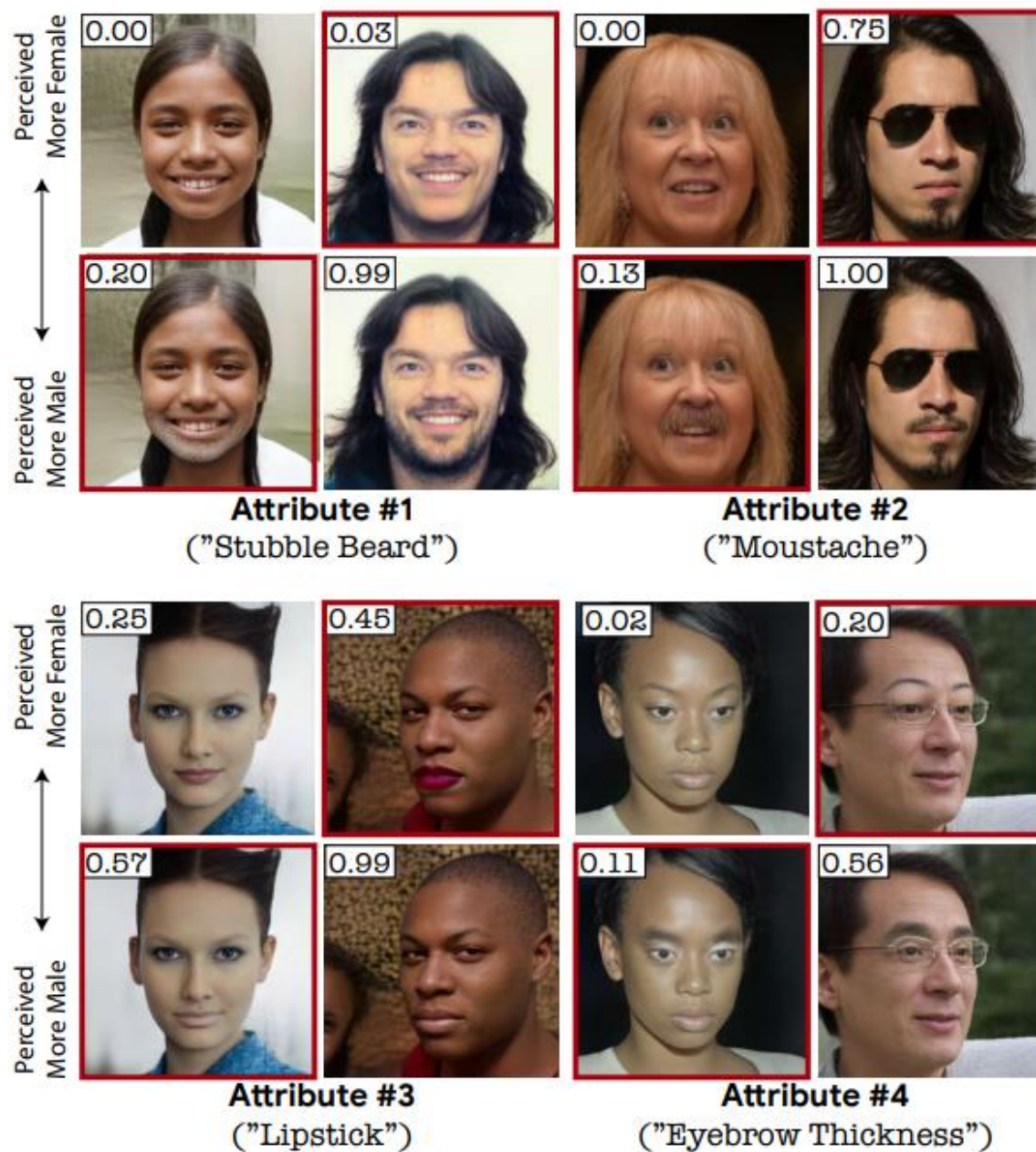
$$Loss = \mathcal{L}_{adv} + \mathcal{L}_{reg} + \mathcal{L}_{rec} + \mathcal{L}_{cls},$$



Experiment

: Probability of being classified as male

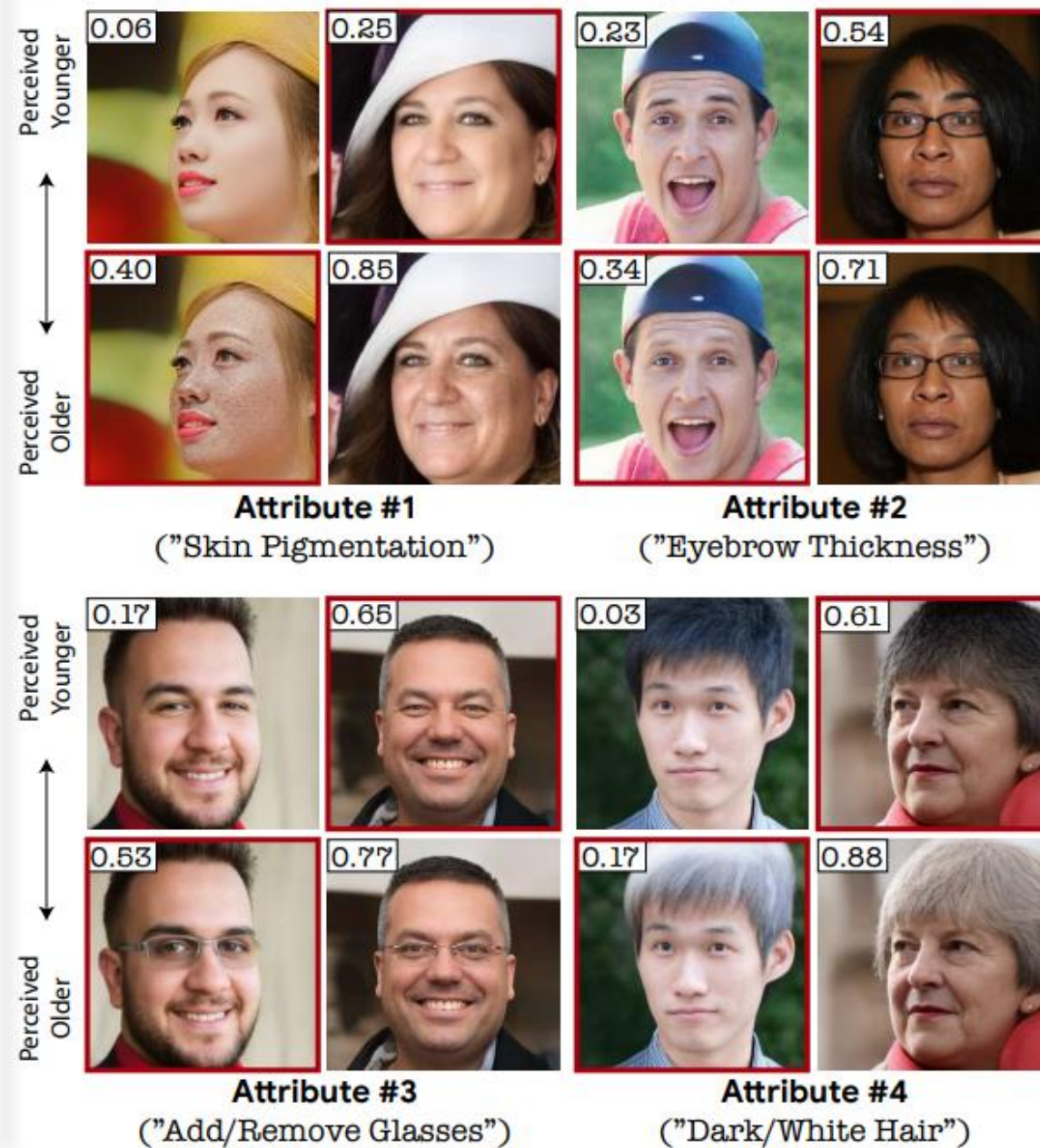
(a) Perceived Gender Classifier



Experiment

: Probability of being classified as old

(b) Perceived Age Classifier



Summary

- Using Classifier-based training of a StyleGAN2 to explain the classification of a given image
- Introducing counterfactual explanations
- Provide meaningful experiment results of Male/Female classifier or Old/Young classifier

Thank you

CoPhy: Counterfactual Learning of Physical Dynamics



Fabien Baradel
INSA-Lyon,
LIRIS



Natalia Neverova
Facebook AI
Research



Julien Mille
INSA-Val de Loire,
LIFAM



Greg Mori
Simon Fraser University,
Borealis AI



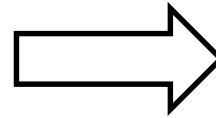
Christian Wolf
INSA-Lyon, LIRIS

Presentation: Jeongho Park (2020-22039)



Reasoning is the Key Ability of Intelligence

- Humans can understand causal relationships between objects. But AI?

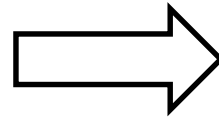


*"The ice cream is melting down due to the **hot weather**"*



Reasoning in Physical(Mechanical) Systems

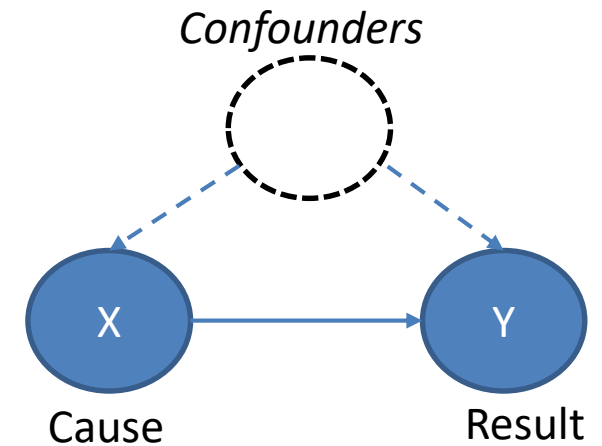
- To predict the future, understanding of physical concepts are necessary



Confounders



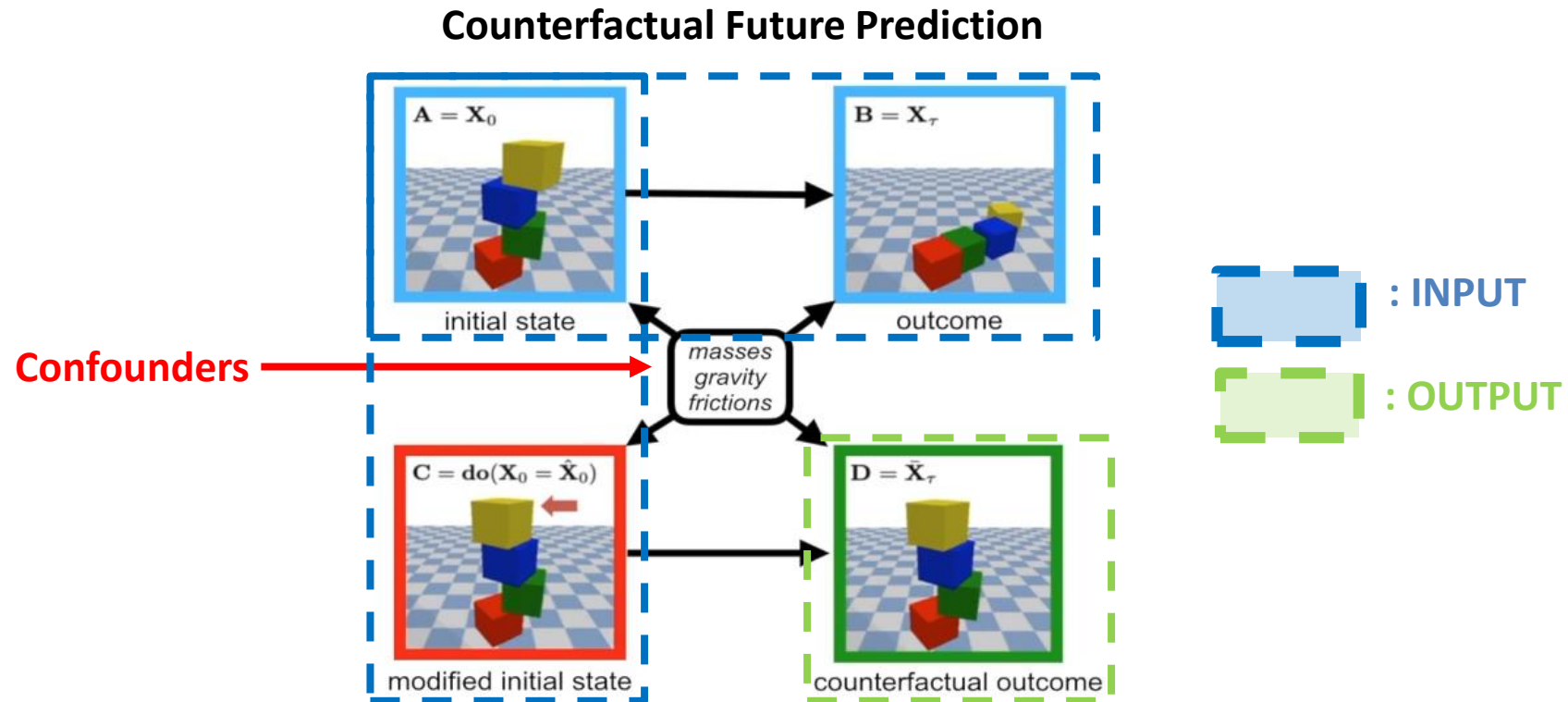
Mass of balls?
Velocity of the cue?
Friction of the floor?



Counterfactual Reasoning

- **Counterfactual Reasoning**

- To predict the **effect of the Modification**(of init. state) based on the given observations without explicitly observing the effect of the modification on data.
- The **Confounder** values have to be learned in order to solve Counterfactual Reasoning problems well.



CoPhyNet – Proposed Network

- **CoPhy Dataset**

- “Counterfactual Physics(CoPhy) Dataset”

- **CoPhyNet**

- **Input:**

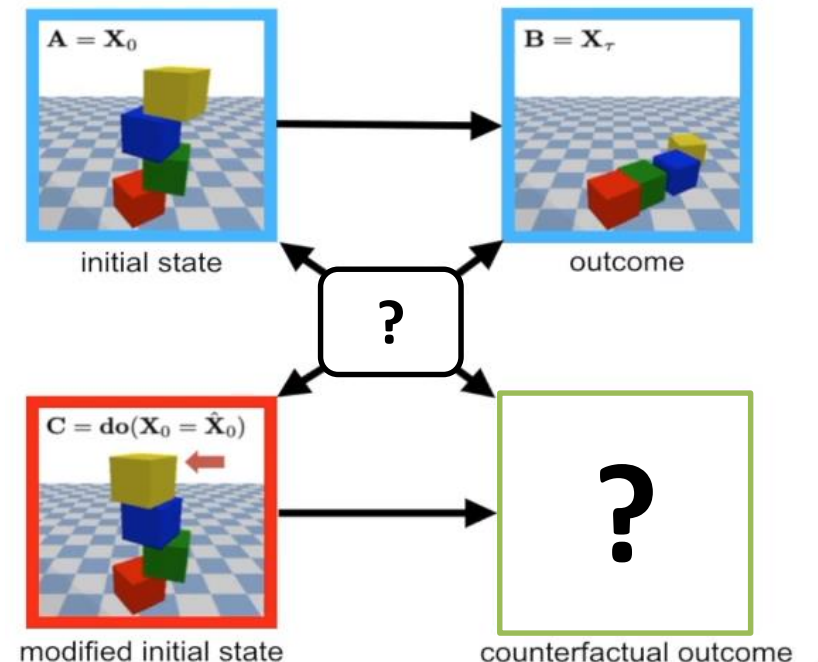
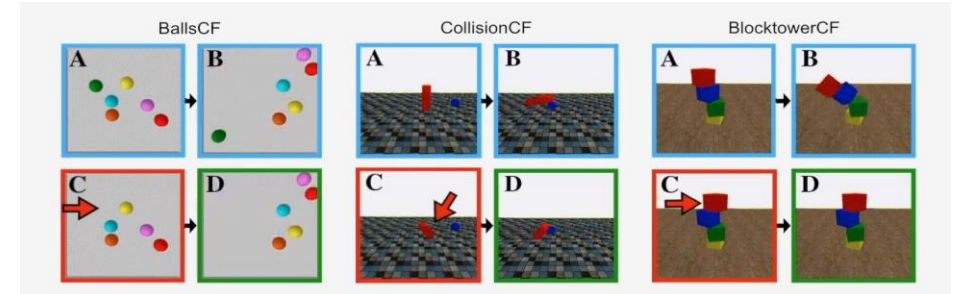
- Observed Sequence ($X = \{X_0, \dots, X_\tau\}$)
- Modified Initial State (\bar{X}_0)
 - Ex) Object displacement, removal, etc.

- **Output:**

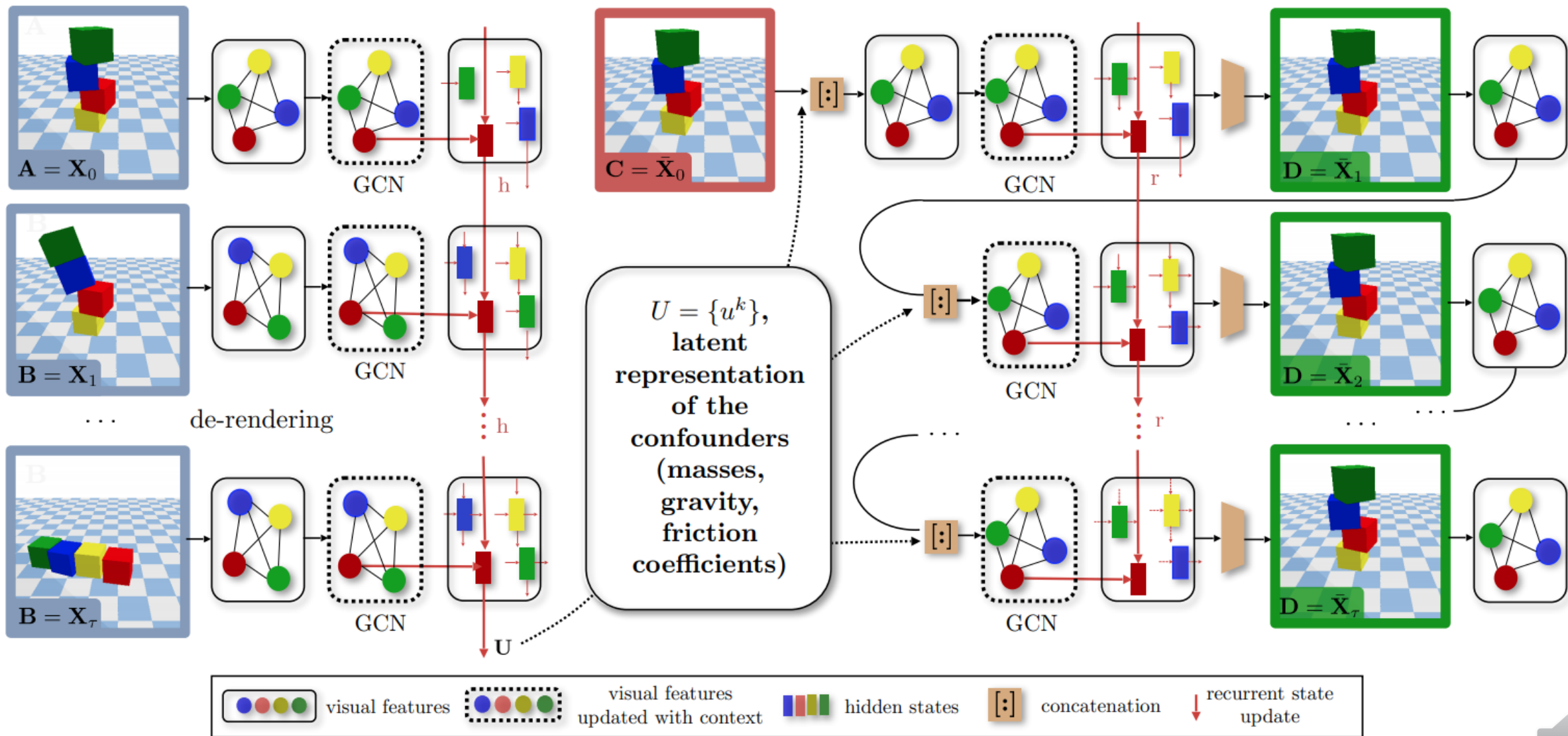
- Counterfactual Sequence Prediction ($\{\bar{X}_1, \dots, \bar{X}_\tau\}$)

- **No Direct supervision** provided for the Confounders

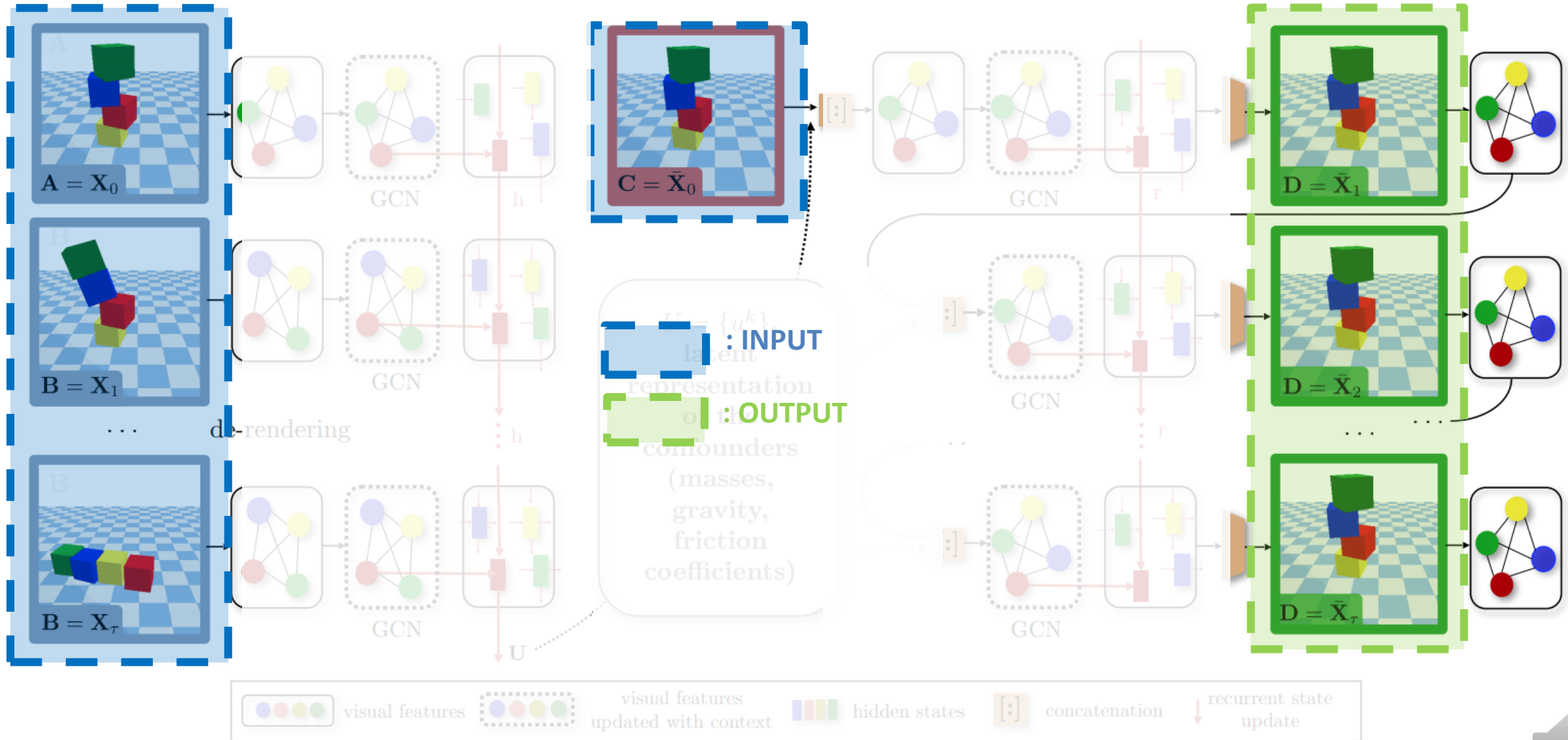
- The network implicitly learns the confounder values.



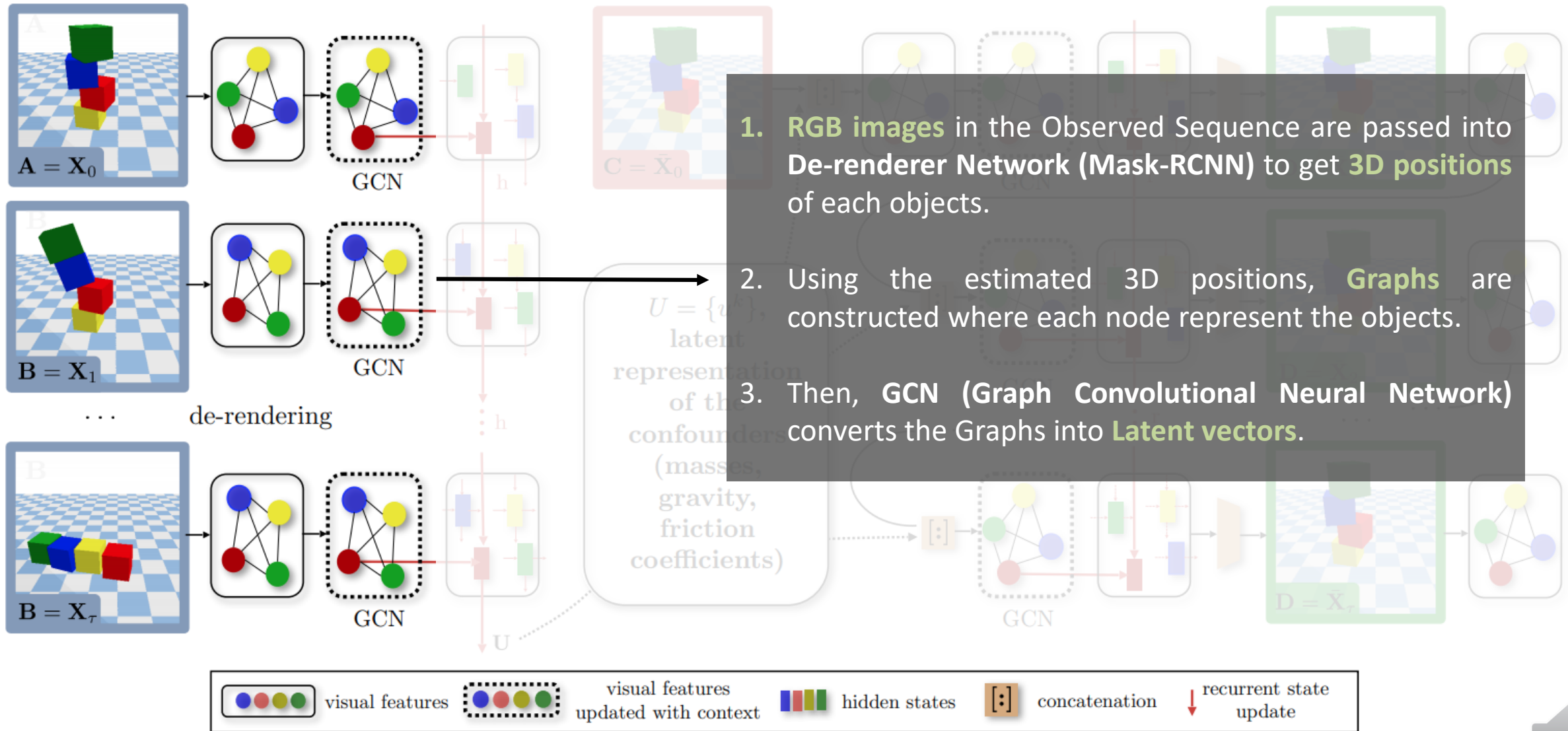
Overall Structure of CoPhyNet



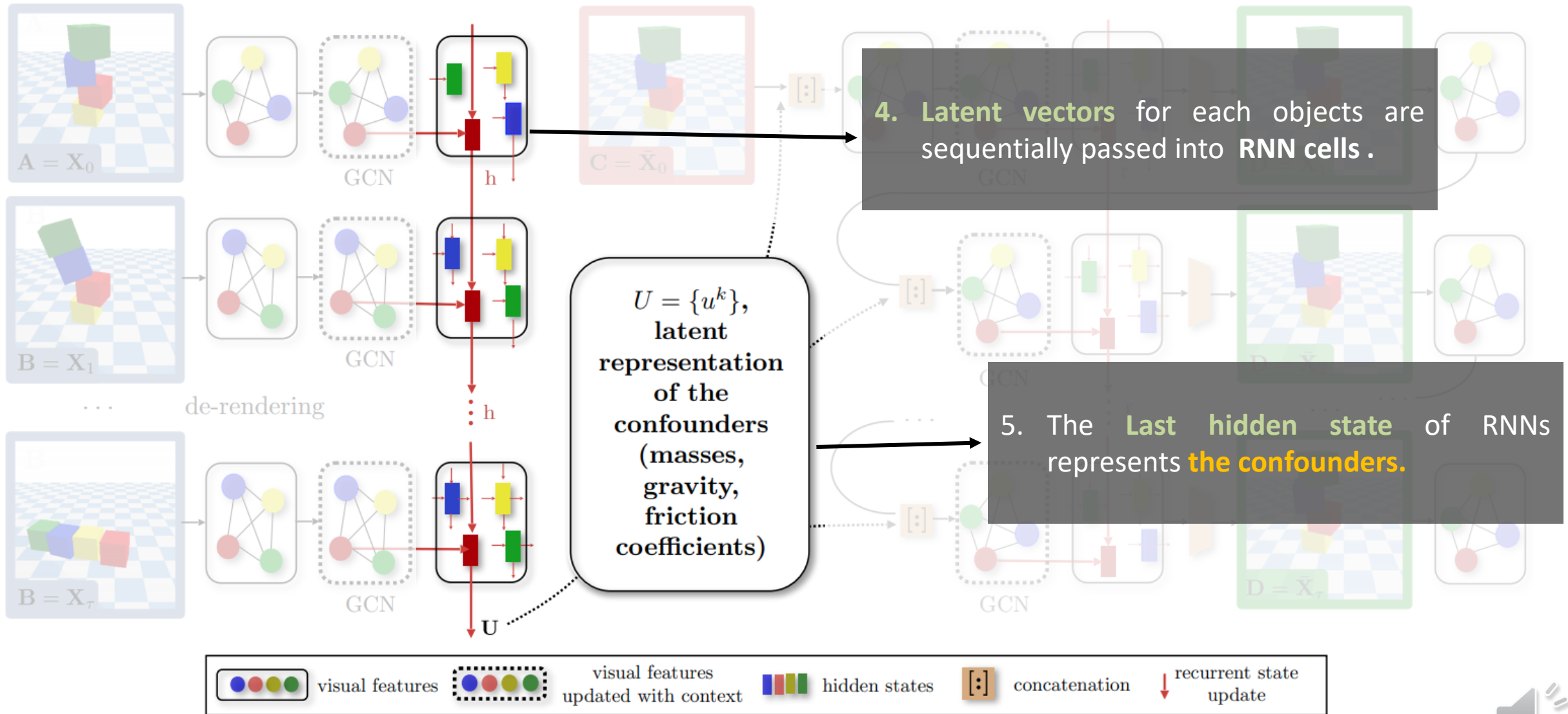
Overall Structure of CoPhyNet



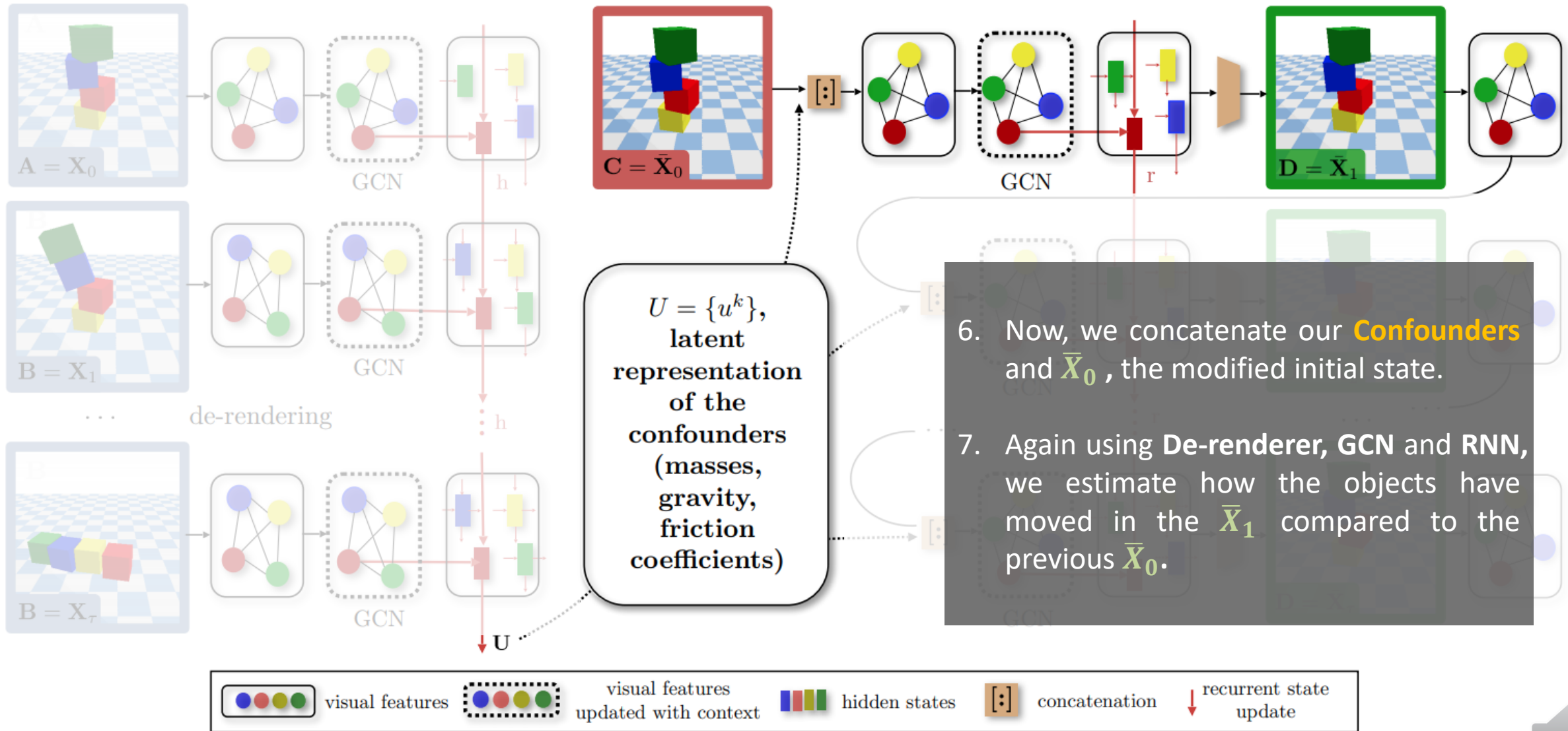
How does CoPhyNet work



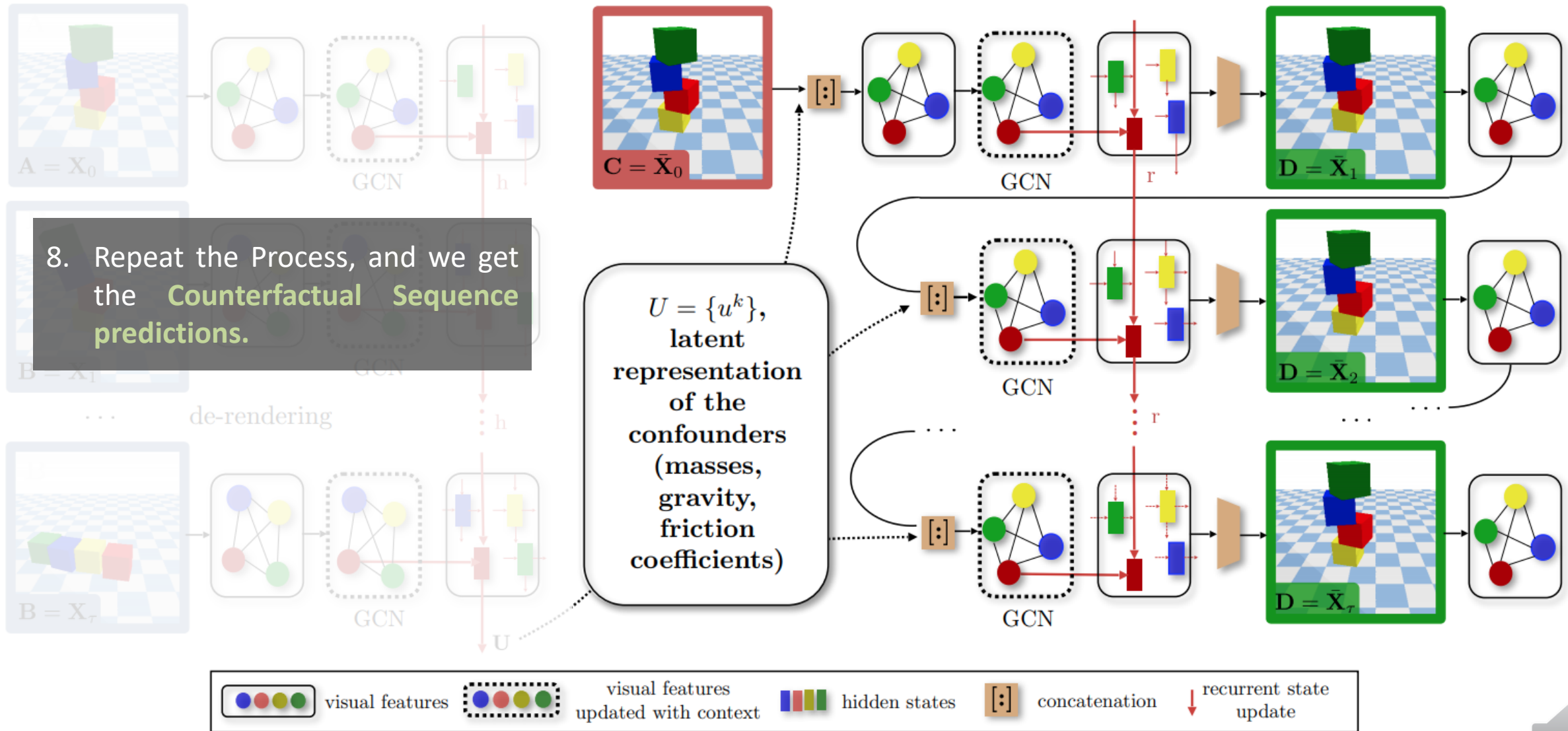
How does CoPhyNet work



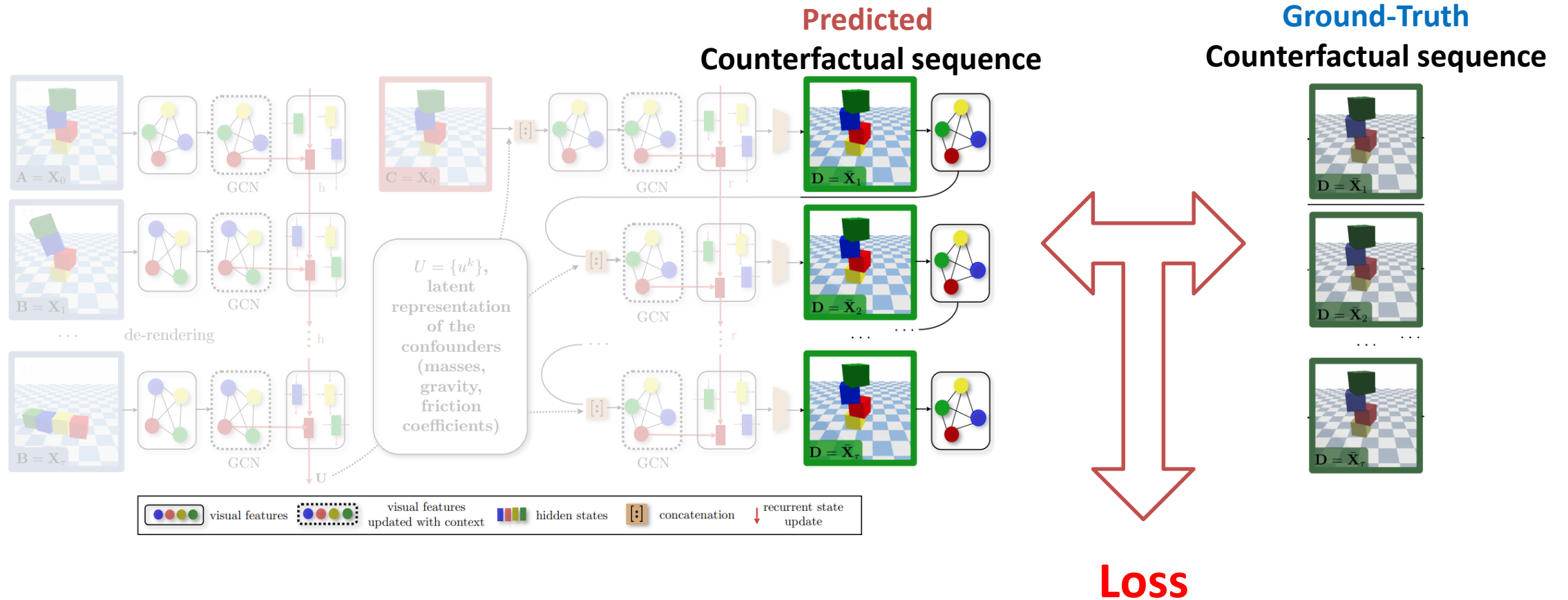
How does CoPhyNet work



How does CoPhyNet work



How to Train CoPhyNet

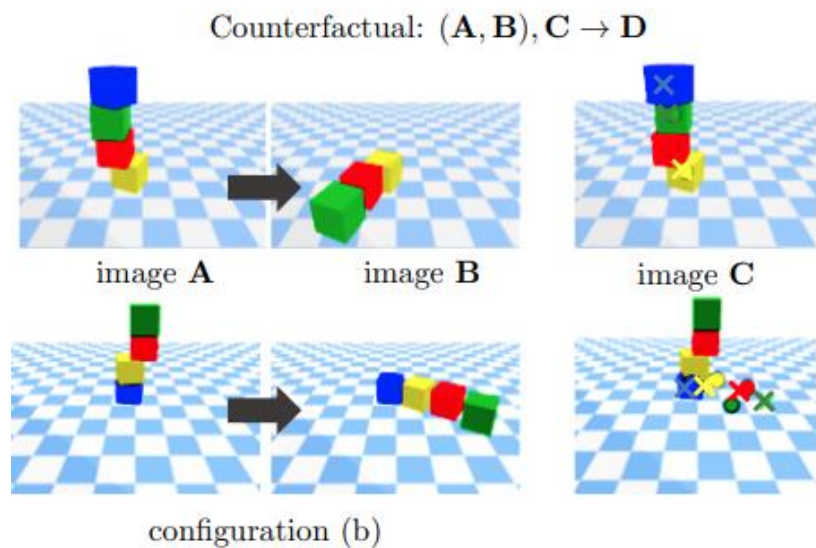


❖ Note that no supervision was provided for confounder values



Experiments

- Comparison between **CoPhyNet** and baselines (**Mean Squared Error**)



O: Ground-Truth position
X: predicted position

Train→Test	Copy C	Copy B	IN	NPE	CoPhyNet	<i>IN sup.</i>
all→all	4.370	0.665	0.701	0.697	0.173	0.332
sphere→cylinder	4.245	0.481	0.715	0.710	0.220	0.435
cylinder→sphere	4.571	0.932	0.720	0.699	0.152	0.586

Table 4: **CollisionCF**: MSE on 3D pose average over time. *IN sup.* methods in the last column exploit the ground truth confounder quantities as input and thus is not directly comparable (still showing inferior performance).



Thank you!



End-to-End Semi-Supervised Object Detection with Soft Teacher

화학생물공학부 박준수



Data matters.

- Obtaining labels can be a bottleneck



Semi-Supervised Learning



→ Labeled data



→ Unlabeled data



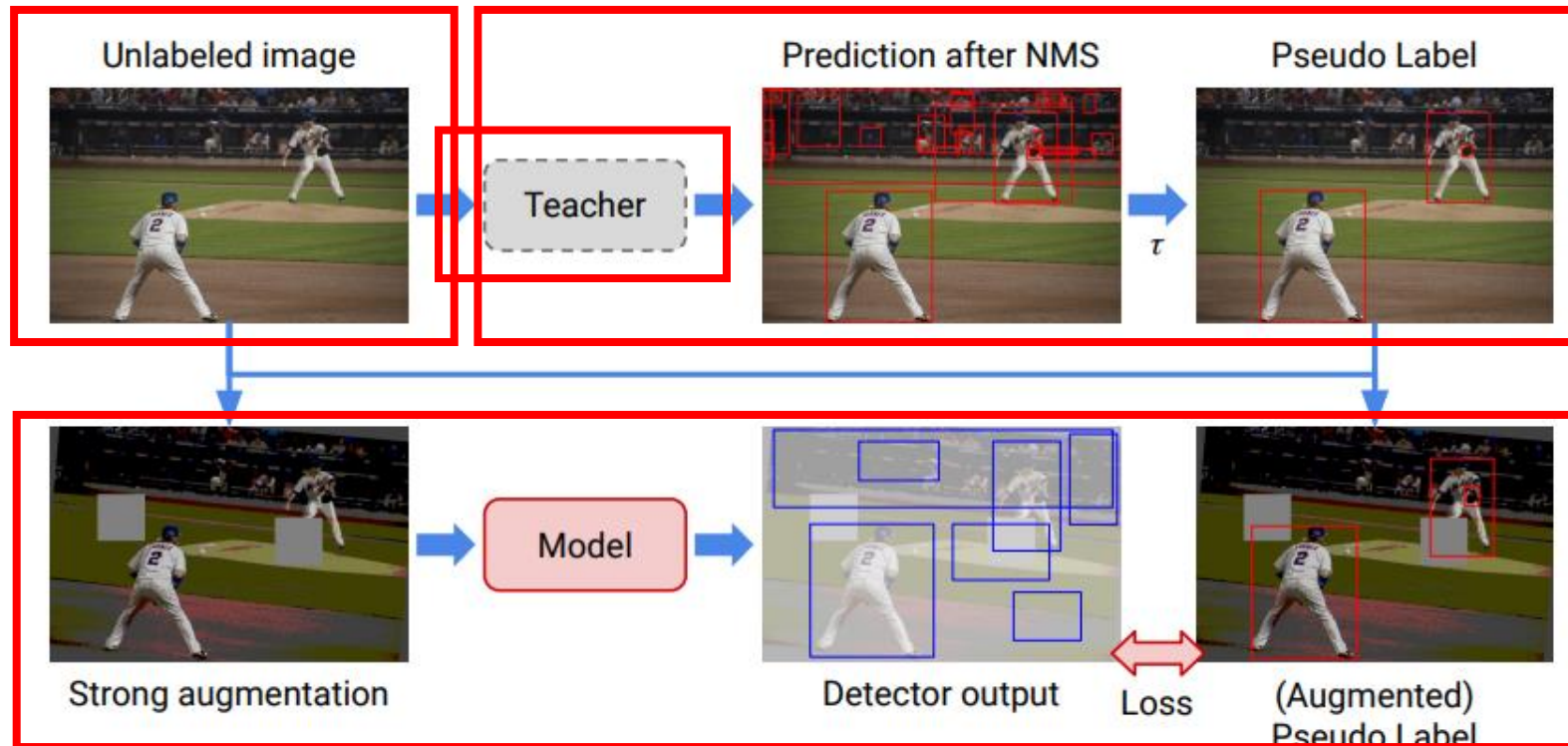
Contribution

- End-to-End Pseudo-Labeling Framework
 - ✓ Teacher gets better and better as the training goes on
- Soft Teacher
 - ✓ Evaluate reliability of background box class during training
- Box Jittering
 - ✓ Evaluate reliability of box regression during training
- State-of-the-art
 - ✓ Soft teacher + Swin-L achieved mAP 61.3 in COCO test-dev set

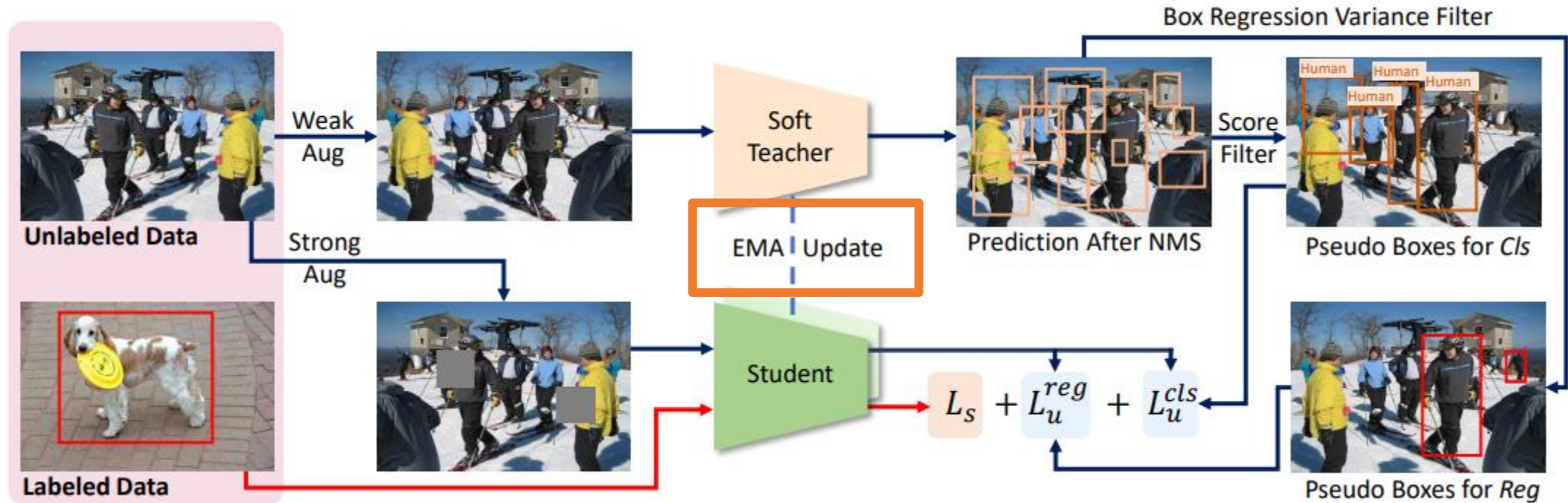


Semi-Supervised Learning

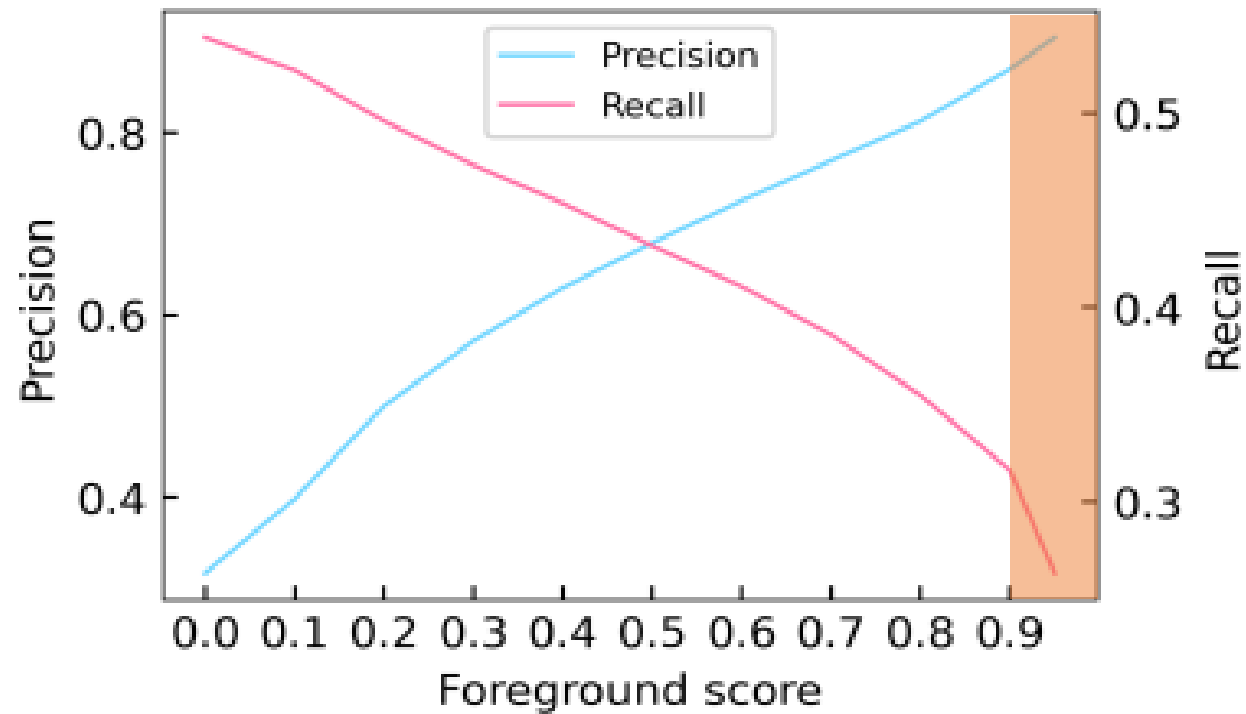
- Consistency based methods
 - ✓ The consistency based methods leverage the unlabeled images to construct a regularization loss which encourages different perturbations of a same image to produce similar predictions
- Pseudo-labeling based methods
 - ✓ The pseudo-label approaches annotate unlabeled images with pseudo labels by an initially trained classification model, and the detector is refined by these pseudo labeled images



End-to-End Pseudo-Labeling Framework

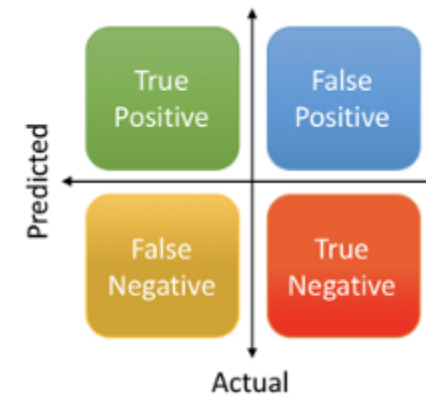


Soft Teacher



$$\text{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

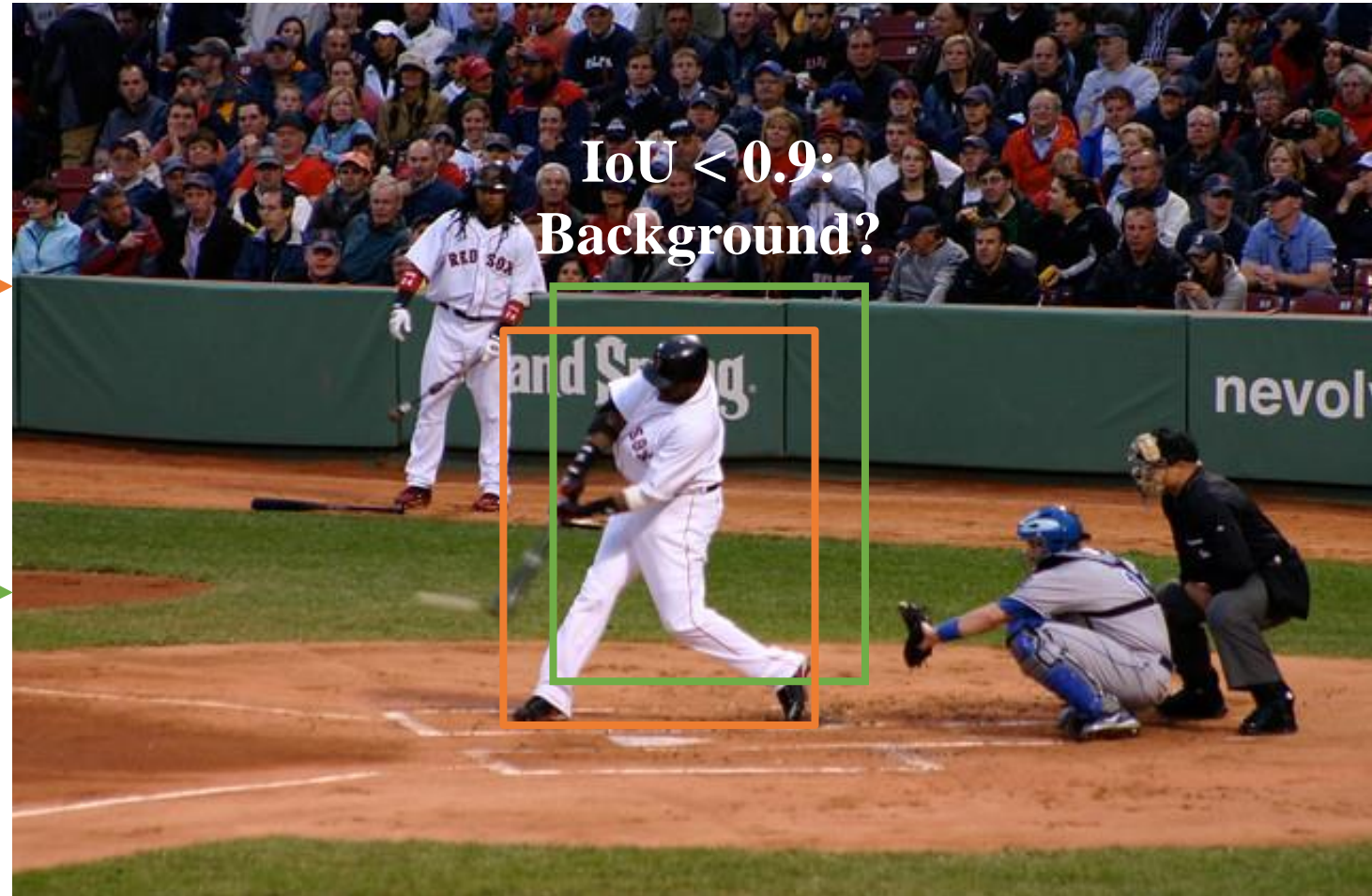
$$\text{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$



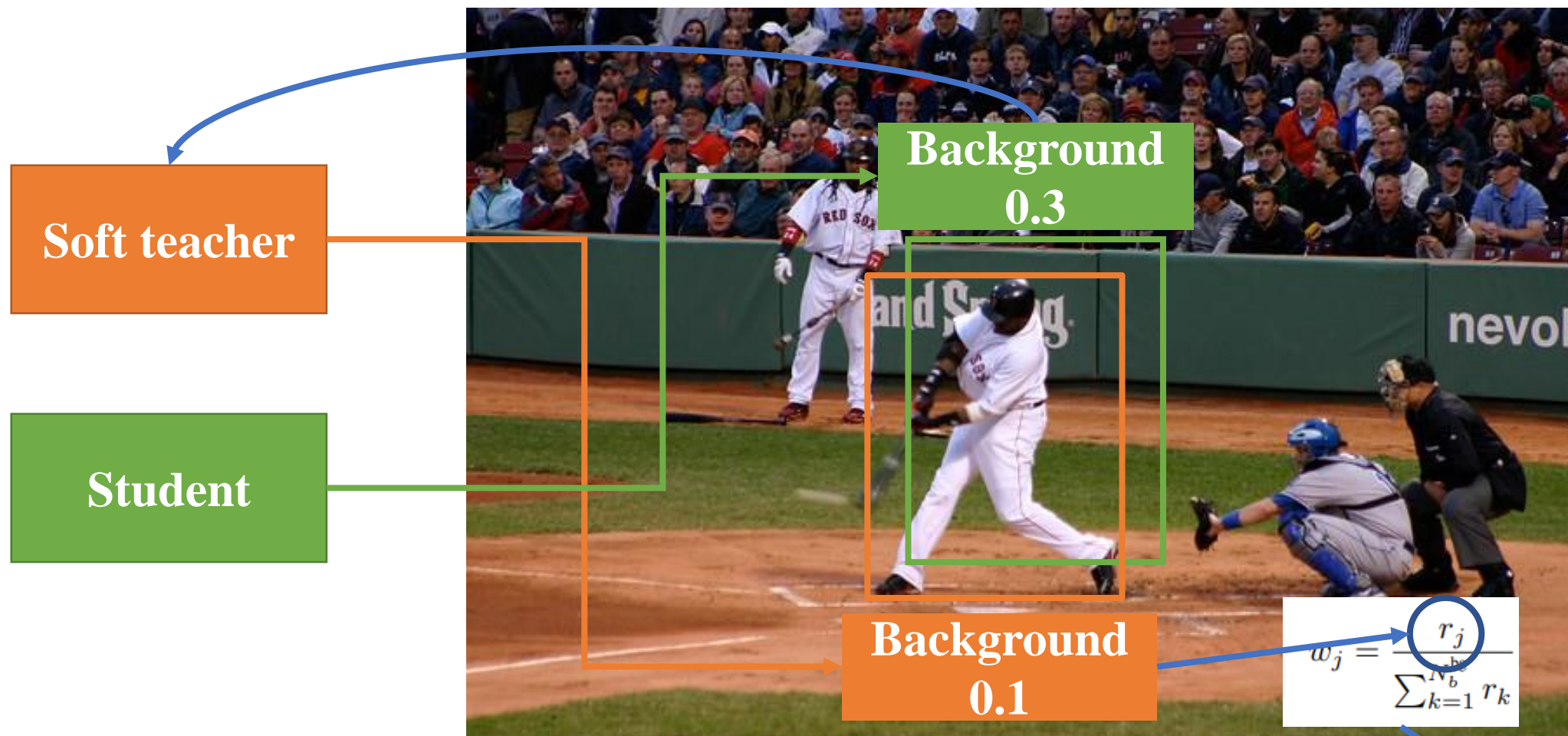
Soft Teacher

Soft teacher

Student

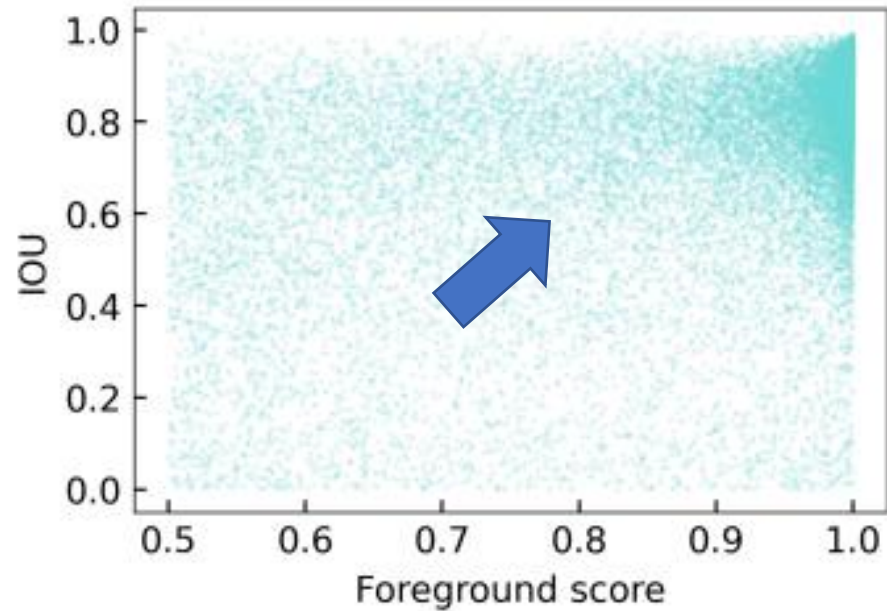


Soft Teacher

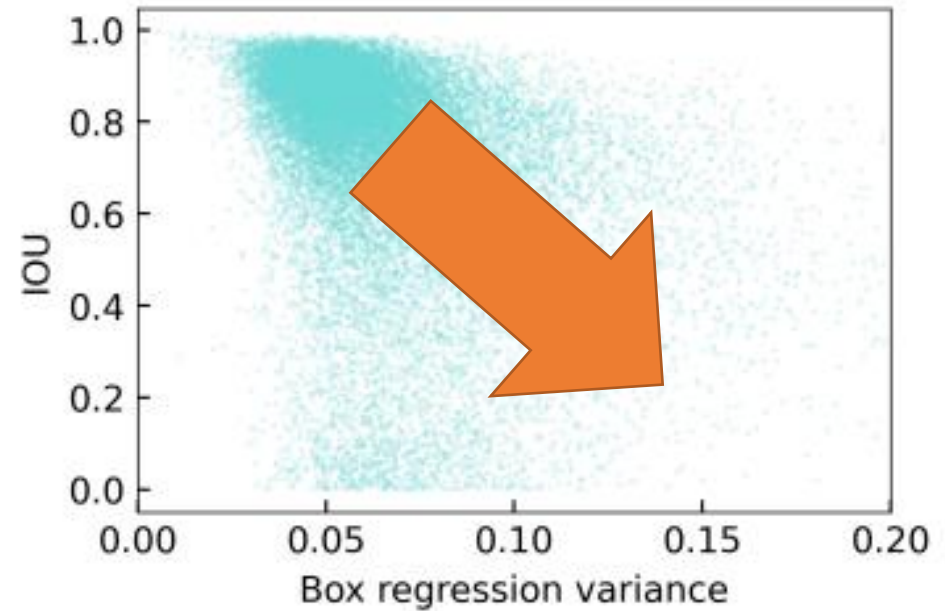


$$\mathcal{L}_u^{\text{cls}} = \frac{1}{N_b^{\text{fg}}} \sum_{i=1}^{N_b^{\text{fg}}} l_{\text{cls}}(b_i^{\text{fg}}, \mathcal{G}_{\text{cls}}) + \sum_{j=1}^{N_b^{\text{bg}}} w_j l_{\text{cls}}(b_j^{\text{bg}}, \mathcal{G}_{\text{cls}}),$$

Box Jittering



Weak positive correlation



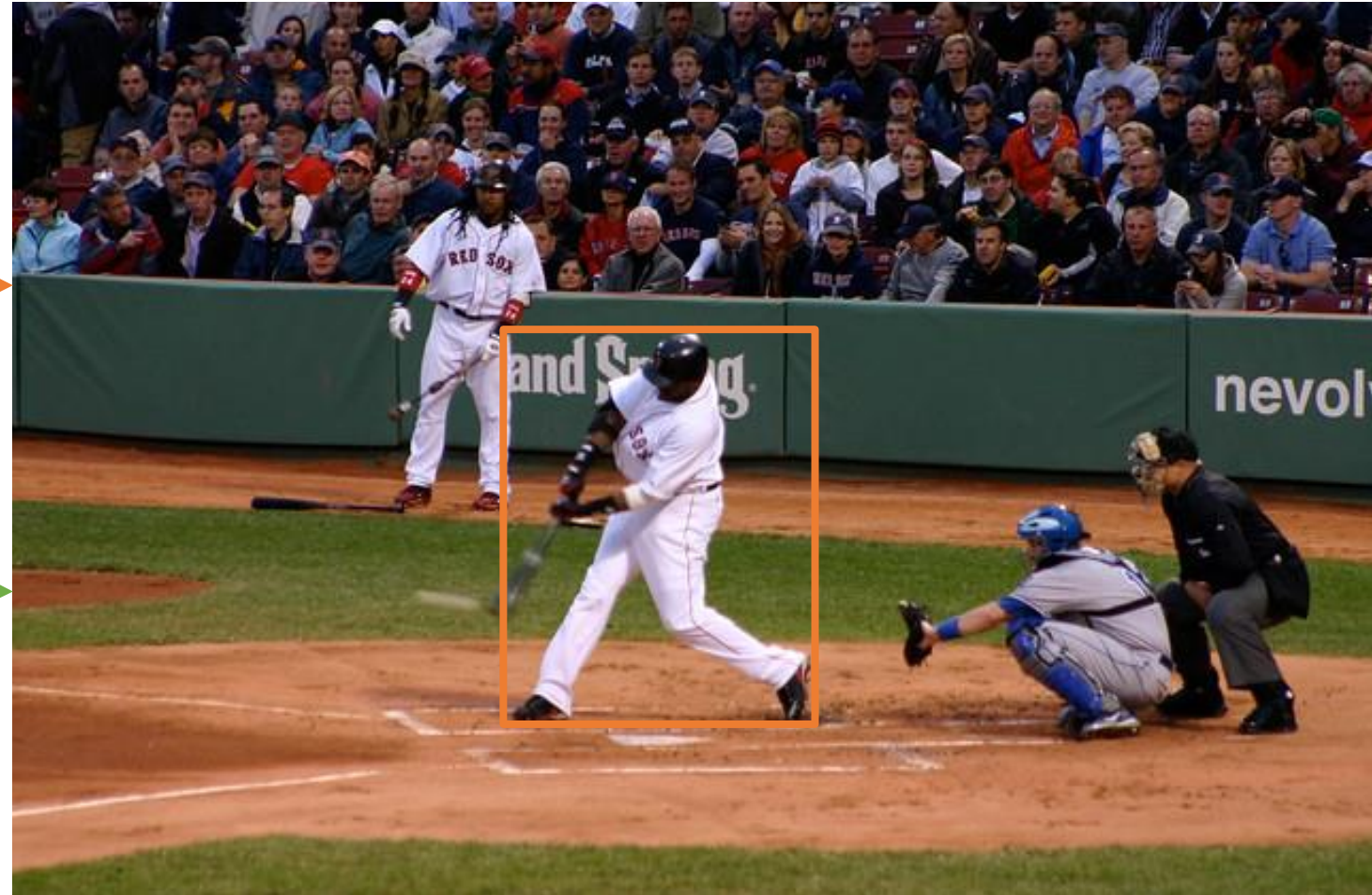
Strong negative correlation



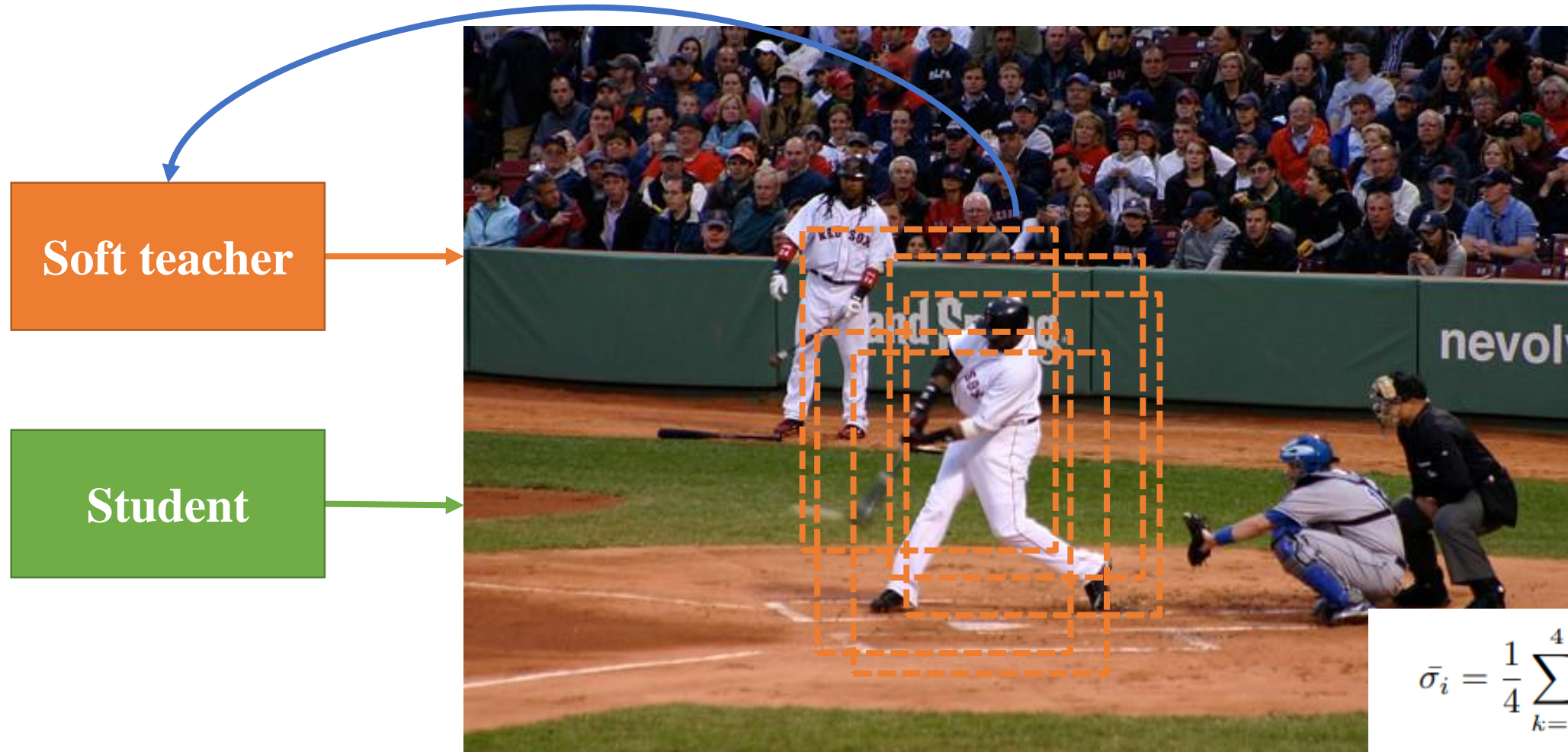
Box Jittering

Soft teacher

Student



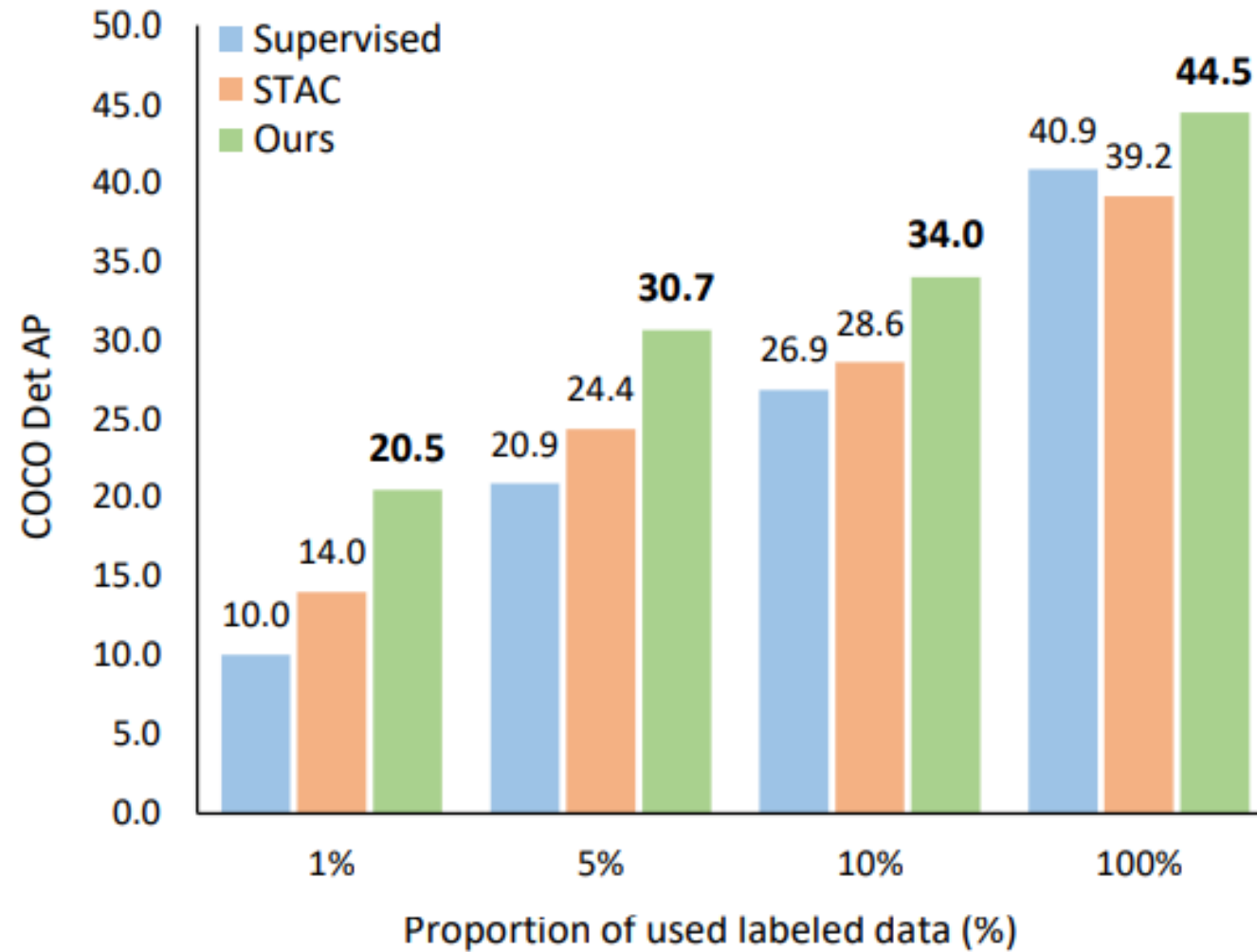
Box Jittering



$$\bar{\sigma}_i = \frac{1}{4} \sum_{k=1}^4 \hat{\sigma}_k,$$

$$\hat{\sigma}_k = \frac{\sigma_k}{0.5(h(b_i) + w(b_i))},$$

State-of-the-art



Contribution

- End-to-End Pseudo-Labeling Framework
 - ✓ Teacher gets better and better as the training goes on
- Soft Teacher
 - ✓ Evaluate reliability of background box class during training
- Box Jittering
 - ✓ Evaluate reliability of box regression during training
- State-of-the-art
 - ✓ Soft teacher + Swin-L achieved mAP 61.3 in COCO test-dev set



THANK YOU!



Fast and Accurate Model Scaling

Facebook AI Research

Jun Young Park



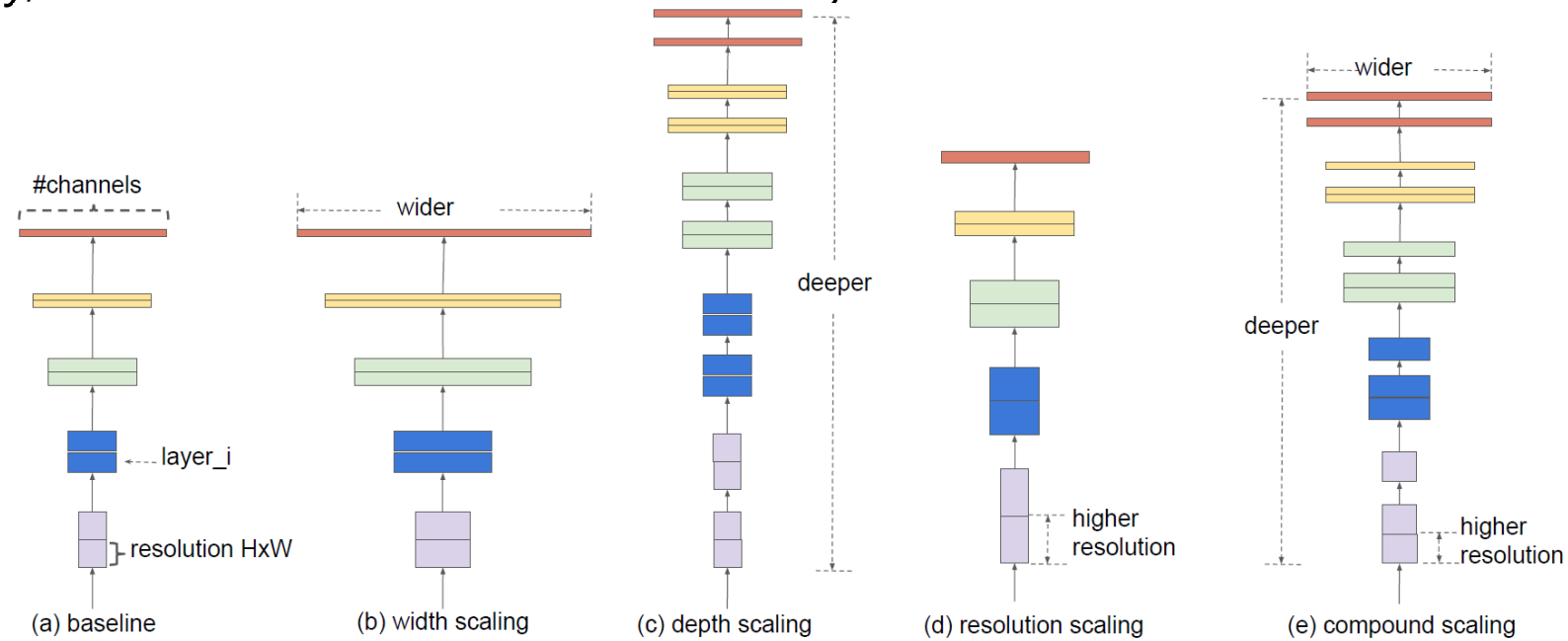
SEOUL
NATIONAL
UNIVERSITY



Background

- Model Scaling

- 기존의 Baseline network를 이용하여 Model의 크기를 키우는 것
- 대표적으로 Depth(Layer 개수), Width(Filter 개수), Resolution(Input image 해상도) 을 조정
- Memory, 연산량 등이 소모되지만 Accuracy 향상을 얻을 수 있다

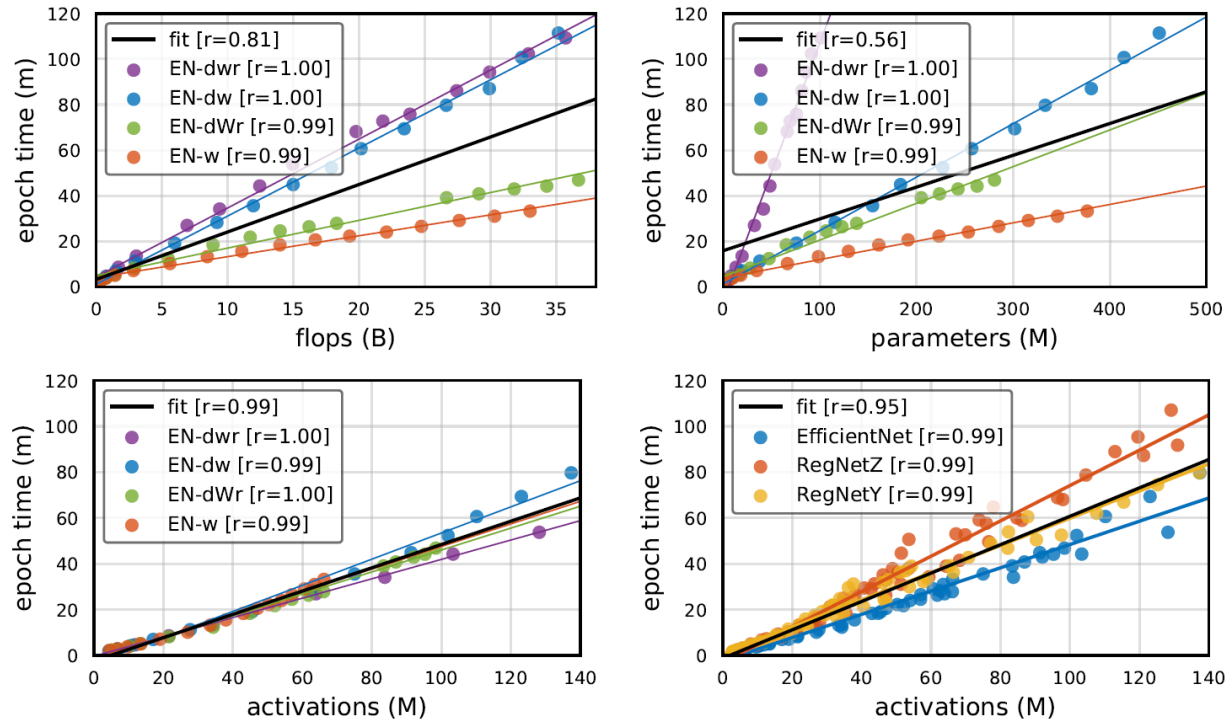


Related Work

- AutoML(Neural Architecture Search)
 - Neural Network의 Architecture와 Hyperparameter를 자동으로 최적화
 - 특정 setting(Dataset, Flop regime)에 맞는 Single model을 찾기 때문에 한계가 존재한다
 - Low or medium compute regimes에 적합
- EfficientNet(2019)
 - width, depth, resolution 를 동시에 고려하여 키우는 Compound scaling 기법을 제안
 - Accuracy에 중점
- RegNet: Designing Network Design Spaces(2020)
 - 좋은 성능을 가진 모델을 찾기 위한 Design space를 Design하는 방법을 제안
 - 논문에서 제안된 RegNet을 Model scaling에 활용함

Key point I

- Runtime은 Activations 와 큰 상관관계가 있다.

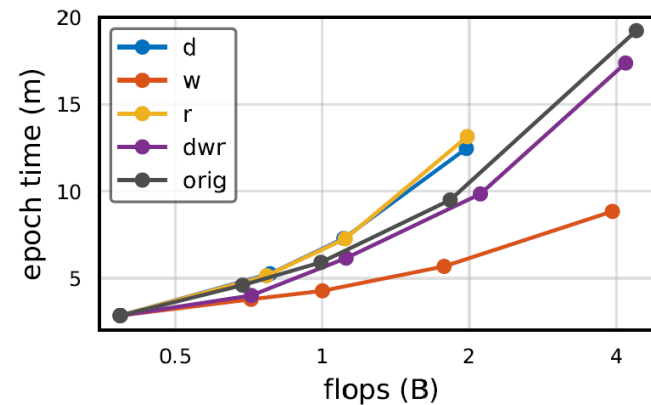
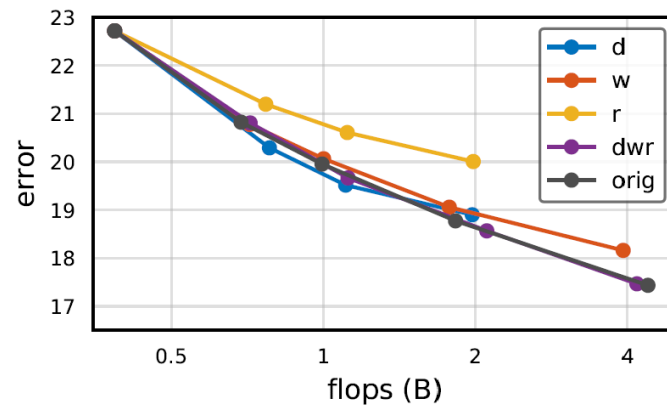


- Flops, parameters는 하나의 모델에 여러 scaling 방법을 사용했을 때, runtime을 예측하기 어렵다
- Activations는 Scaling 방법, Model architecture를 변경해도 Runtime을 예측하기 쉽다

Key point II

- Width scaling⁰이 Activations 증가를 최소화한다

dim	scaling			flops (f)	params (p)	acts (a)
none	d	w	r	dw^2r^2	dw^2	dwr^2
d	sd	w	r	sdw^2r^2	sdw^2	$sdwr^2$
w	d	\sqrt{sw}	r	sdw^2r^2	sdw^2	$\sqrt{sdwr^2}$
r	d	w	\sqrt{sr}	sdw^2r^2	$1dw^2$	$sdwr^2$



- 동일한 Flops에 대해 Width scaling⁰이 Activations, Runtime 증가를 최소로 한다.
- Accuracy 측면에서 Compound Scaling(dwr)이 좋다.

Key point III

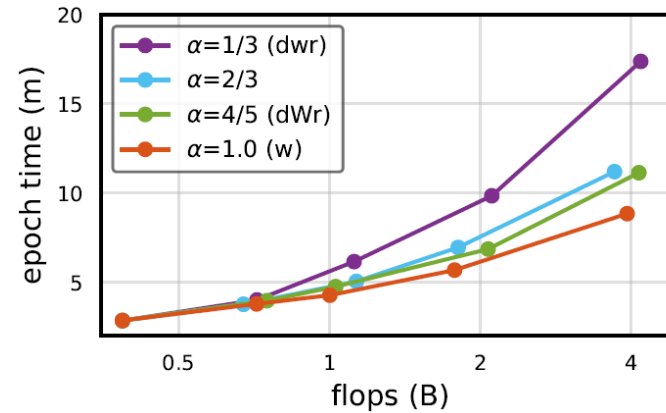
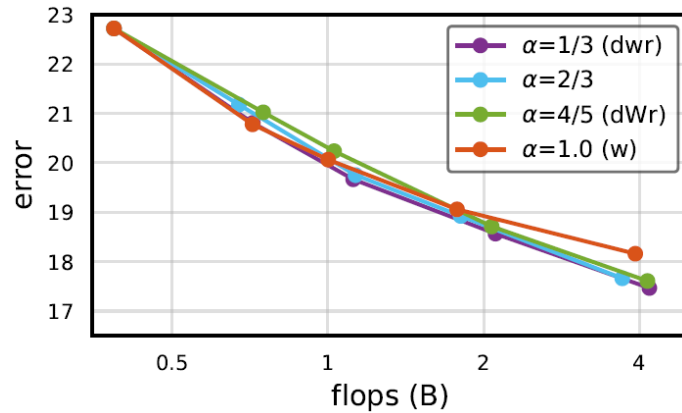
- Width 중점으로 depth, resolution도 같이 증가시키는 Fast Scaling 을 제안한다

dims	α	e_d	e_w	e_r	f	p	a
	α	$\frac{1-\alpha}{2}$	α	$\frac{1-\alpha}{2}$	s	$s^{\frac{1+\alpha}{2}}$	$s^{\frac{2-\alpha}{2}}$
dr	0	0.5	0	0.5	s	$s^{0.50}$	$s^{1.00}$
dwr	1/3	1/3	1/3	1/3	s	$s^{0.67}$	$s^{0.83}$
dWr	0.8	0.1	0.8	0.1	s	$s^{0.90}$	$s^{0.60}$
w	1	0	1	0	s	$s^{1.00}$	$s^{0.50}$

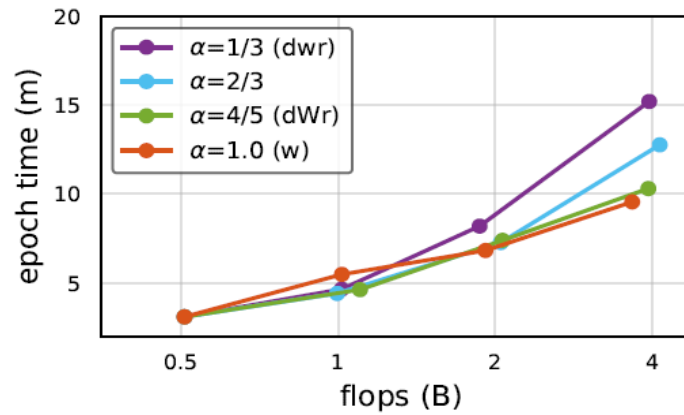
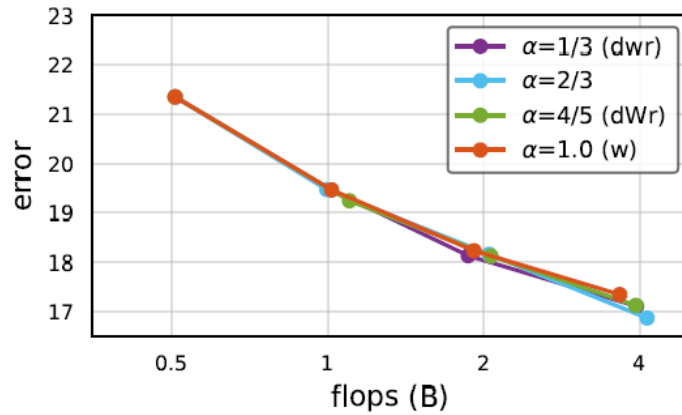
- 가중치 $\alpha = 1$ 일 때, width만 scaling하며 가장 작은 activations 를 가진다
- 가중치 $\alpha = 0$ 일 때, depth와 resolution을 scaling하며 가장 큰 activations 를 가진다.
- 가중치 α 가 1에 가까울수록 Fast scaling이라 하며, 0.8을 default(dWr)로 잡는다.

Results

- dWr scaling yields good accuracy and speed



Fast Scaling: EfficientNet



Fast Scaling: RegNetZ

Results

- dWr scaling yields good accuracy and speed

	flops (B)	params (M)	acts (M)	time (min)	1×	2×	4×
ResNet50 [9]	4.1	25.6	11.3	3.5	22.0 \pm 0.12	21.0 \pm 0.08	20.5 \pm 0.07
ResNeXt50 [32]	4.2	25.0	14.6	5.8	20.8 \pm 0.06	19.9 \pm 0.16	19.5 \pm 0.05
EfficientNet-B4 [31]	4.4	19.3	49.5	19.2	18.0 \pm 0.05	17.4 \pm 0.07	17.3 \pm 0.06
RegNetY-4GF	4.1	22.4	14.5	7.7	18.8 \pm 0.04	18.0 \pm 0.07	17.7 \pm 0.09
Scaling by dWr → RegNetZ-4GF	4.0	28.1	24.3	11.7	17.5 \pm 0.09	17.0 \pm 0.12	16.9 \pm 0.04
EfficientNet-B0→4GF	4.1	36.1	29.2	11.1	18.4 \pm 0.11	17.7 \pm 0.07	17.4 \pm 0.11
RegNetY-500MF→4GF	4.1	36.2	13.3	7.2	19.1 \pm 0.07	18.6 \pm 0.09	18.3 \pm 0.06
RegNetZ-500MF→4GF	4.0	41.1	19.4	10.5	17.7 \pm 0.07	17.2 \pm 0.07	17.0 \pm 0.05
EfficientNet-B0→16GF	16.2	122.8	61.8	25.8	17.4 \pm 0.08	16.8 \pm 0.09	–
RegNetY-500MF→16GF	16.2	112.7	29.4	17.8	17.8 \pm 0.18	17.2 \pm 0.06	16.9 \pm 0.10
RegNetY-4GF→16GF	15.5	72.3	30.7	16.4	17.3 \pm 0.09	16.8 \pm 0.11	16.6 \pm 0.03
RegNetZ-500MF→16GF	16.2	134.8	42.6	29.4	16.6 \pm 0.04	16.1 \pm 0.06	16.1 \pm 0.07
RegNetZ-4GF→16GF	15.9	95.3	51.3	33.2	16.5 \pm 0.05	16.0 \pm 0.10	16.0 \pm 0.05

Results

- dWr scaling yields good accuracy and speed

	flops (B)	params (M)	acts (M)	time (min)	schedule		
					1×	2×	4×
ResNet50 [9]	4.1	25.6	11.3	3.5	22.0 \pm 0.12	21.0 \pm 0.08	20.5 \pm 0.07
ResNeXt50 [32]	4.2	25.0	14.6	5.8	20.8 \pm 0.06	19.9 \pm 0.16	19.5 \pm 0.05
EfficientNet-B4 [31]	4.4	19.3	49.5	19.2	18.0 \pm 0.05	17.4 \pm 0.07	17.3 \pm 0.06
RegNetY-4GF	4.1	22.4	14.5	7.7	18.8 \pm 0.04	18.0 \pm 0.07	17.7 \pm 0.09
RegNetZ-4GF	4.0	28.1	24.3	11.7	17.5 \pm 0.09	17.0 \pm 0.12	16.9 \pm 0.04
EfficientNet-B0→4GF	4.1	36.1	29.2	11.1	18.4 \pm 0.11	17.7 \pm 0.07	17.4 \pm 0.11
RegNetY-500MF→4GF	4.1	36.2	13.3	7.2	19.1 \pm 0.07	18.6 \pm 0.09	18.3 \pm 0.06
RegNetZ-500MF→4GF	4.0	41.1	19.4	10.5	17.7 \pm 0.07	17.2 \pm 0.07	17.0 \pm 0.05
EfficientNet-B0→16GF	16.2	122.8	61.8	25.8	17.4 \pm 0.08	16.8 \pm 0.09	–
RegNetY-500MF→16GF	16.2	112.7	29.4	17.8	17.8 \pm 0.18	17.2 \pm 0.06	16.9 \pm 0.10
RegNetY-4GF→16GF	15.5	72.3	30.7	16.4	17.3 \pm 0.09	16.8 \pm 0.11	16.6 \pm 0.03
RegNetZ-500MF→16GF	16.2	134.8	42.6	29.4	16.6 \pm 0.04	16.1 \pm 0.06	16.1 \pm 0.07
RegNetZ-4GF→16GF	15.9	95.3	51.3	33.2	16.5 \pm 0.05	16.0 \pm 0.10	16.0 \pm 0.05

Scaling by dWr →

Conclusion

- Runtime에 가장 큰 영향을 미치는 요인은 Activation 수이다.
 - Width scaling은 Activations 증가를 최소화한다
- Width를 중심으로 Scaling하는 Fast scaling 제안
 - Accuracy & Runtime 모두 좋은 성능을 보인다

GS-WGAN: A Gradient-Sanitized Approach for Learning Differentially Private Generators

Dingfan Chen
NeurIPS 2020

Presentator: Chanwoong Park

DP?

DP?



왜 그들은

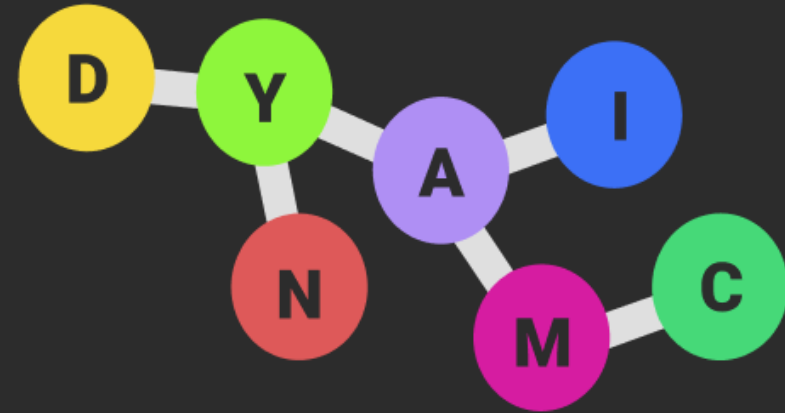
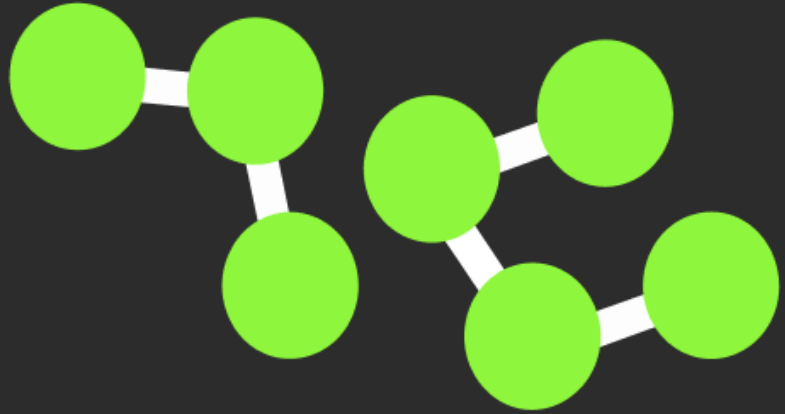
탈영병이 되었나

디피

넷플릭스 시리즈

8월 27일 공개 | NETFLIX

DP?



Divide and Conquer vs Dynamic Programming

DP?

Differential Privacy: Quantify the level of privacy

DP?

Definition

A randomized mechanism \mathcal{M} with range \mathcal{R} is (ϵ, δ) -DP if

$$\Pr[\mathcal{M}(S) \in \mathcal{O}] \leq e^\epsilon \cdot \Pr[\mathcal{M}(S') \in \mathcal{O}] + \delta$$

holds for any subset of outputs $\mathcal{O} \in \mathcal{R}$ and for any adjacent datasets S and S' , where S and S' differ from each other with only one training example.

Just Example



직원 A: 전기과에 재벌집
자식이 3명이 있어요!

교수 B: 전기과에 재벌
이 2명 있다는군...



친구: 자퇴함.



나: 내 친구가... 재벌?!



Just Example

$$S = \{\text{전기과 학생들}\} \quad N(S) = 100$$

$$S_1 = S - \{\text{친구}\}$$

$$S_2 = S - \{\text{나}\}$$

$$P(\text{재벌}|S) = \frac{3}{100}$$

1%

$$P(\text{재벌}|S_2) = \frac{3}{99}$$

50%

$$P(\text{재벌}|S_1) = \frac{2}{99}$$

친구가 수상함!

If one sample makes a big difference,
privacy is not guaranteed.

Conversely, for privacy to be guaranteed,
probabilities should be bound.

Differential Privacy

Definition

A randomized mechanism \mathcal{M} with range \mathcal{R} is (ϵ, δ) -DP if

$$\Pr[\mathcal{M}(S) \in \mathcal{O}] \leq e^\epsilon \cdot \Pr[\mathcal{M}(S') \in \mathcal{O}] + \delta$$

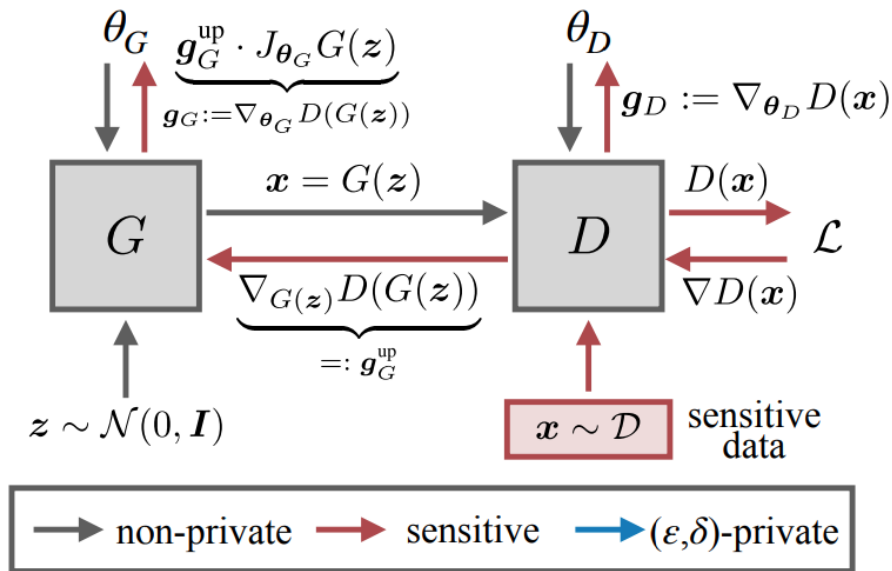
holds for any subset of outputs $\mathcal{O} \in \mathcal{R}$ and for any adjacent datasets S and S' , where S and S' differ from each other with only one training example.

Mathematical Background

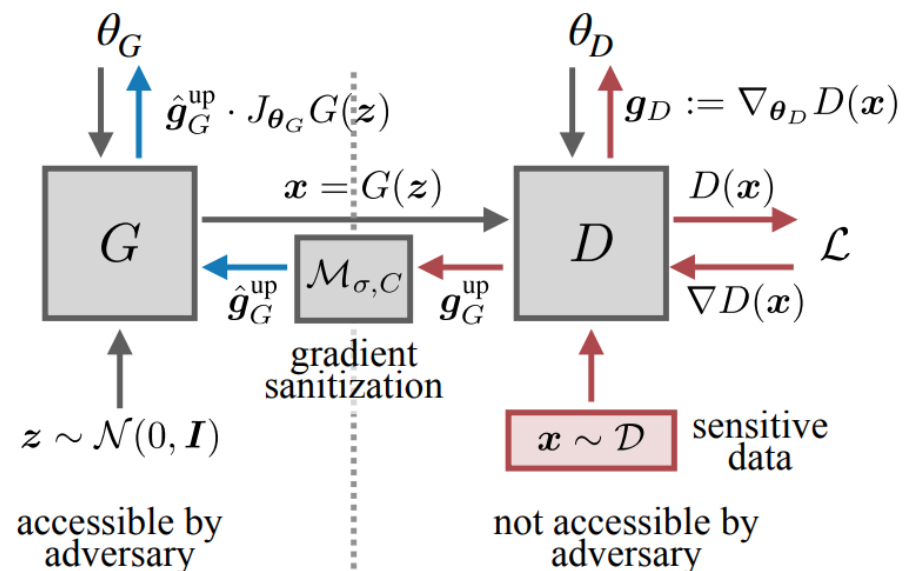
- ▶ (λ, ϵ) -RDP is $(\epsilon + \frac{\log 1/\delta}{\lambda-1}, \delta)$ -DP.
- ▶ The Gaussian Mechanism $\mathcal{M}_\sigma(x) = f(x) + \mathcal{N}(0, \sigma^2 I)$ is $(\lambda, \frac{\lambda \Delta_2 f^2}{2\sigma^2})$ -RDP.
- ▶ If \mathcal{M}_i is (λ, ϵ_i) -RDP, the composition $\mathcal{M}_1 \circ \dots \circ \mathcal{M}_k$ is also $(\lambda, \sum_i \epsilon_i)$ -RDP.
- ▶ If \mathcal{M} is (ϵ, δ) -DP, $F \circ \mathcal{M}$ is also (ϵ, δ) -DP for any function F .

GS-WGAN

- We want the DP-Generator.
- Just add Gaussian noise to the gradient of the generator in WGAN.
- The discriminator is trained normally.



(a) Vanilla GAN
(Without privacy barrier)



(b) GS-WGAN
(Ours, with privacy barrier)

Experiment


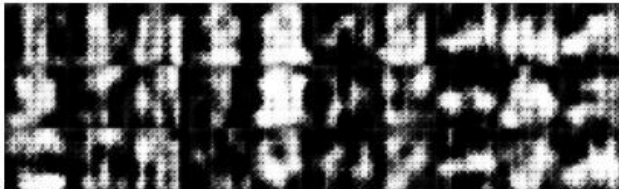

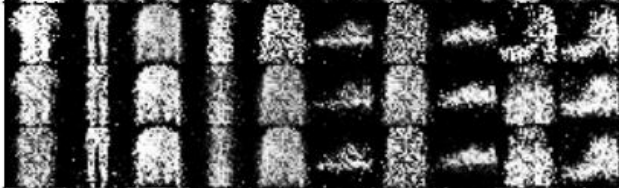
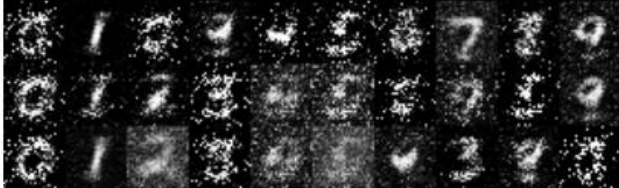
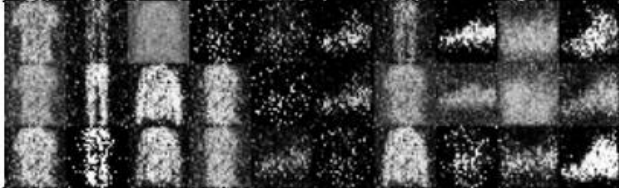

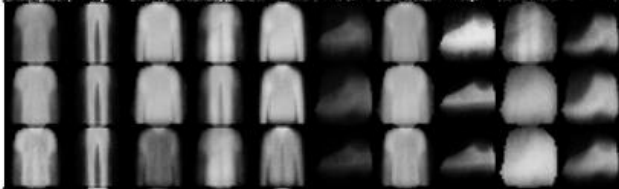
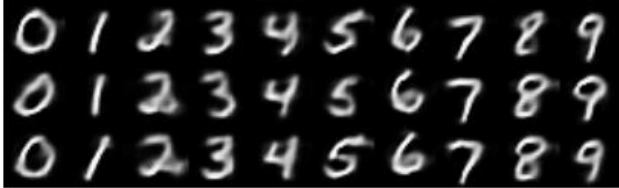
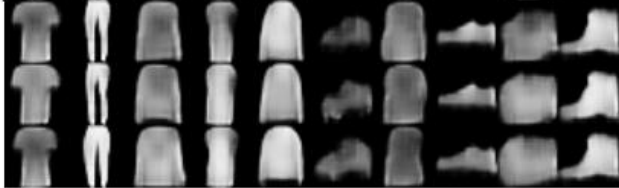
Method	MNIST	Fashion-MNIST
G-PATE		
DP-SGD GAN		
DP-Merf		
DP-Merf AE		
Ours		

Figure 3: Generated samples with $(\epsilon, \delta) = (10, 10^{-5})$

Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness

*Jeremiah Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax Weiss,
Balaji Lakshminarayanan*

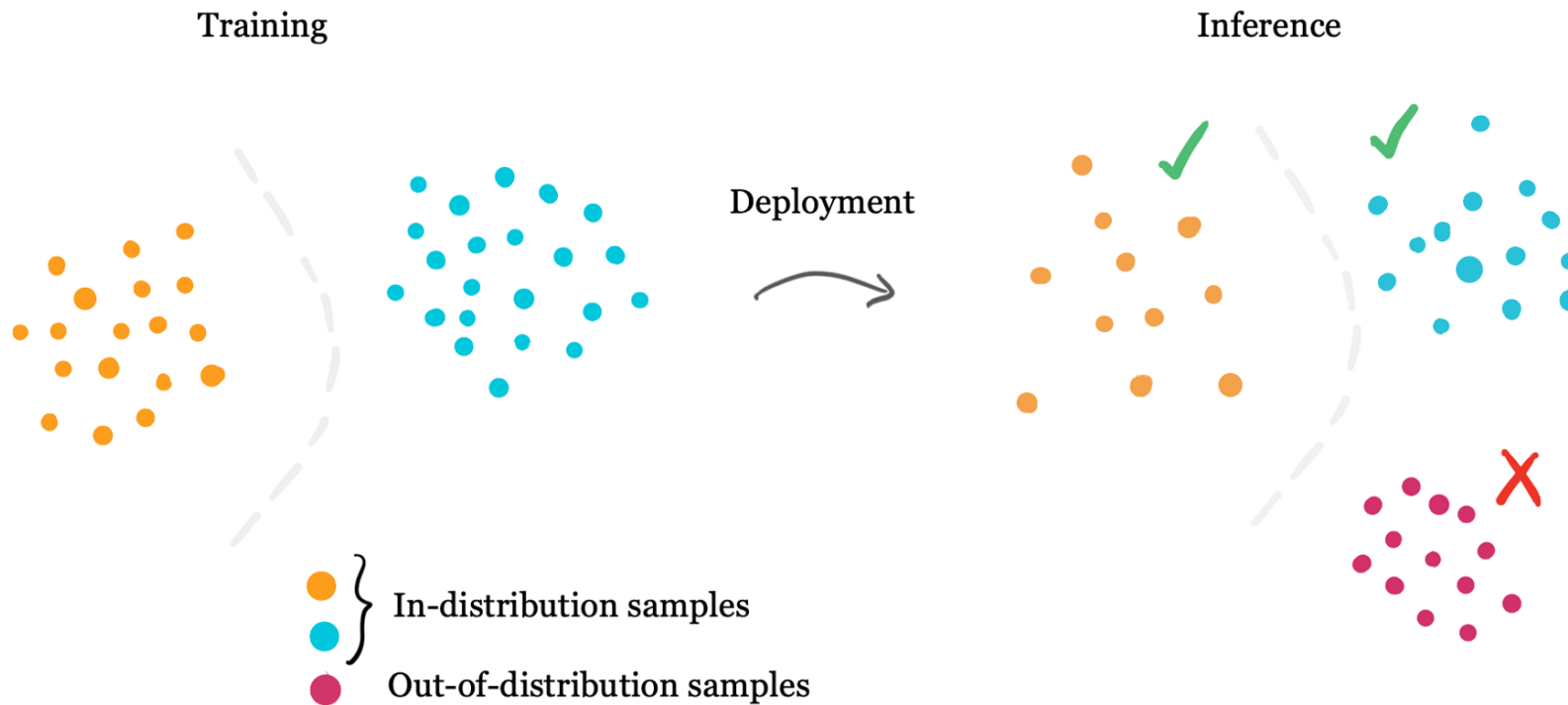
NeurIPS 2020

발표자: 박충현



OOD(Out-Of Distribution) Detection

- 학습하지 않은 데이터 감지
 - 학습한 domain과 학습하지 않은 domain 구별



이미지 출처: <https://medium.com/geekculture/out-of-distribution-detection-in-medical-ai-b638b385c2a3>

OOD Detection

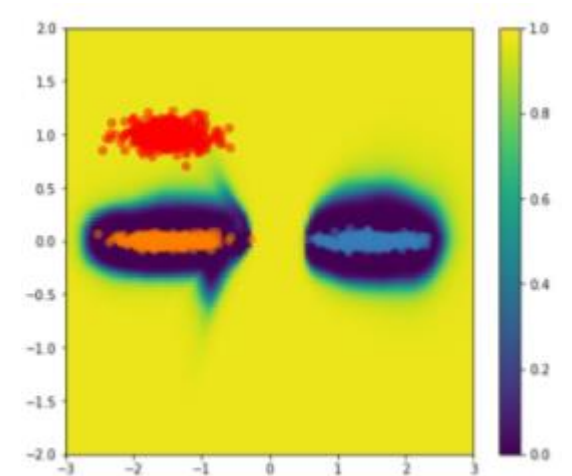
- Optimal Training for OOD

- 입력에 대한 uncertainty 계산
- In-domain data:는 학습한대로 처리
- Out-of-distribution data: uniform distribution output

$$p(y|\mathbf{x}) = p(y|\mathbf{x}, \mathbf{x} \in \mathcal{X}_{\text{IND}}) * p^*(\mathbf{x} \in \mathcal{X}_{\text{IND}}) + p_{\text{uniform}}(y|\mathbf{x}, \mathbf{x} \notin \mathcal{X}_{\text{IND}}) * p^*(\mathbf{x} \notin \mathcal{X}_{\text{IND}})$$

- Proposed method: SNGP

- 입력, 학습 데이터 사이 거리 계산 가능
- 거리 기반 uncertainty



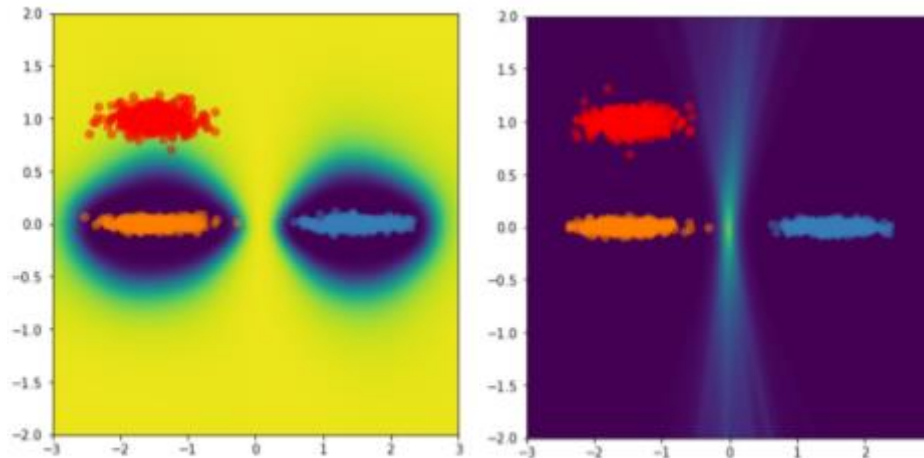
(e) SNGP (Ours)

Input distance awareness

- 입력 데이터와 전체 학습 데이터 사이 거리를 통해 uncertainty를 수치화
 - 거리에 따라 단조 증가

$$u(\mathbf{x}) = v(d(\mathbf{x}, \mathcal{X}_{\text{IND}}))$$

- DNN의 distance-awareness property
 - Typical discriminative classifier: not input distance aware
 - decision boundary과의 거리 기준
 - Gaussian Process (RBF kernel): input distance aware



(a) Gaussian Process

(b) Deep Ensemble

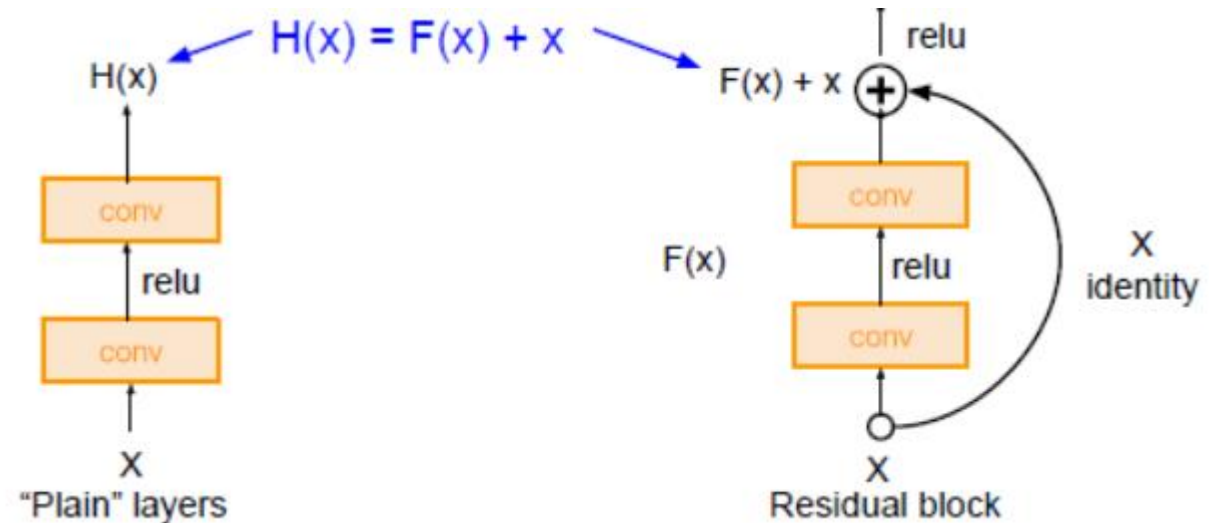
Spectral-normalized Neural Gaussian Process (SNGP)

- Distance-awareness for residual-based DNN

- Residual block: 많이 사용되는 구조
 - ResNet
 - Transformer

- How?

- Distance aware output layer
- Distance preserving hidden layers



이미지 출처: <https://itrepo.tistory.com/36>

Distance aware output layer

- Original concept

- Typical dense output layer를 GP layer로 교체
- Prior distribution

$$g_{N \times 1} \sim MVN(\mathbf{0}_{N \times 1}, \mathbf{K}_{N \times N}), \text{ where } \mathbf{K}_{i,j} = \exp(-\|h_i - h_j\|_2^2/2)$$

- Posterior distribution

- $p(g)$: prior
- $p(D|g)$: data likelihood for classification (cross-entropy loss)

$$p(g|\mathcal{D}) \propto p(\mathcal{D}|g)p(g)$$

- 계산량 과다

- Low-rank approximation for kernel matrix

Distance preserving Hidden layers

- Spectral Normalization

- Residual architecture의 특성

- $h = h_{L-1} \circ \dots \circ h_2 \circ h_1$ where $h_l(\mathbf{x}) = \mathbf{x} + g_l(\mathbf{x})$, $0 < \alpha \leq 1$;
 - 각 hidden layer의 residual output g_l 이 모두 $\|g_l(\mathbf{x}) - g_l(\mathbf{x}')\|_H \leq \alpha \|\mathbf{x} - \mathbf{x}'\|_X$ 이면,
 - h 는 아래 식을 만족하는 distance preserving한 output

$$L_1 * \|\mathbf{x} - \mathbf{x}'\|_X \leq \|h(\mathbf{x}) - h(\mathbf{x}')\|_H \leq L_2 * \|\mathbf{x} - \mathbf{x}'\|_X, \text{ where } L_1 = (1 - \alpha)^{L-1} \text{ and } L_2 = (1 + \alpha)^{L-1}$$

- α 조정

- Residual block: $g_l(\mathbf{x}) = \sigma(\mathbf{W}_l \mathbf{x} + \mathbf{b}_l)$
 - $0 < \alpha \leq 1$ 이기 위한 조건: $\|\mathbf{W}_l\|_2 \leq 1$

- => 학습시 각 step마다 weight matrix에 spectral normalization 적용

Reference

- Liu, Jeremiah Zhe, et al. "Simple and principled uncertainty estimation with deterministic deep learning via distance awareness." *arXiv preprint arXiv:2006.10108* (2020).

Differentiable Convex Optimization Layers

Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, J. Zico Kolter
33rd Conference on Neural Information Processing Systems

Presenter : Taeon Park

Seoul National University
Machine **IN**telligence and **D**ata science Laboratory

2021. 11. 20



M.IN.D Lab

Preliminary

- Convex optimization problem (Lecture slide 5)
 - A certain class of constrained optimization

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \dots, m \\ & Ax = b, \end{array}$$

with variable $x \in \mathbb{R}^n$.
(f_0, \dots, f_m are convex in x)

- Many problems can be formulated as convex optimizations
- **Hard to solve** this problem in general



Preliminary

- Convex optimization problem (Lecture slide 5)
 - A certain class of constrained optimization

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f_0(x; \theta) \\ \text{subject to} & f_i(x; \theta) \leq 0, \quad i = 1, \dots, m \\ & A(\theta)x = b(\theta), \end{array}$$

with variable $x \in \mathbb{R}^n$ and parameters $\theta \in \mathbb{R}^p$.
(f_0, \dots, f_m are convex in x for each θ)

- Many problems can be formulated as convex optimizations
- **Hard to solve** this problem in general

→ This paper provides a **differentiable neural network layer** that solves the **convex optimization problem**



Preliminary

- CVXPY

- Python-embedded, **high level** language for convex optimization
- Transforms **user-friendly form** into **solver-friendly form**
- No connection with **neural networks** before this paper
- e.g.,

$$\underset{x}{\text{minimize}} \quad \sum_{i=1}^n (y_i - x)^2$$

with variable $x \in \mathbb{R}$ and parameters $y \in \mathbb{R}^n$.

The solution map is clearly: $x^* = \frac{1}{n} \sum_{i=1}^n y_i$

And the gradient is as follows: $\nabla_y x^* = \frac{1}{n} \mathbf{1}$



Preliminary

- CVXPY

– e.g.,

$$\underset{x}{\text{minimize}} \quad \sum_{i=1}^n (y_i - x)^2$$

with variable $x \in \mathbb{R}$ and parameters $y \in \mathbb{R}^n$.

```
import cvxpy
import numpy as np
n = 4
```

1. Import packages

```
x = cvxpy.Variable()
y = cvxpy.Parameter(n)
```

2. Define variables & parameters

```
objective = cvxpy.sum_squares(y - x)
constraints = []
```

3. Define objective and constraints

```
problem = cvxpy.Problem(cvxpy.Minimize(objective), constraints)
```

4. Synthesize problem

```
y.value = np.random.randn(n)
```

5. Set parameter values

```
problem.solve()
```

6. Solve problem in one line

```
print('y:', y.value)
print("CVXPY solution:", "%.3f" % x.value)
print("Analytical solution:", "%.3f" % np.mean(y.value))
```

Check the result

```
y: [1.764 0.4 0.979 2.241]
CVXPY solution: 1.346
Analytical solution: 1.346
```

Differentiating through CVXPY

- From CVXPY to CVXPYLAYERS (**proposed**)

– e.g.,

$$\underset{x}{\text{minimize}} \quad \sum_{i=1}^n (y_i - x)^2$$

with variable $x \in \mathbb{R}$ and parameters $y \in \mathbb{R}^n$.

```
import cvxpy
import numpy as np
n = 4

x = cvxpy.Variable()
y = cvxpy.Parameter(n)

objective = cvxpy.sum_squares(y - x)
constraints = []

problem = cvxpy.Problem(cvxpy.Minimize(objective), constraints)

y.value = np.random.randn(n)

problem.solve(requires_grad=True)

print('y:', y.value)
print("CVXPY solution:", "%.3f" % x.value)
print("Analytical solution:", "%.3f" % np.mean(y.value))
```

7. Set gradient w.r.t x

8. Differentiate in one line

```
x.gradient = np.array([1.])
problem.backward()
print("CVXPY gradient:", y.gradient)
print("Analytical gradient:", np.ones(y.size) / n)

CVXPY gradient: [0.25 0.25 0.25 0.25]
Analytical gradient: [0.25 0.25 0.25 0.25]
```

```
y: [1.764 0.4 0.979 2.241]
CVXPY solution: 1.346
Analytical solution: 1.346
```

Differentiating through CVXPY

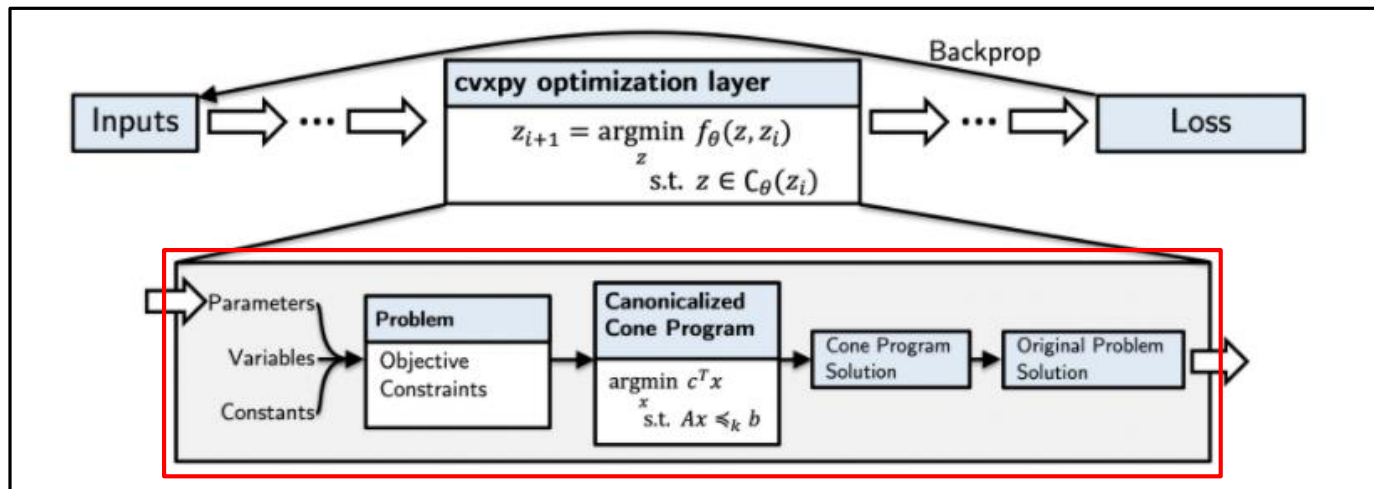
- From CVXPY to CVXPYLAYERS (**proposed**)
 - In the previous slide, we saw the differentiation with CVXPY
 - Convex optimization problems can also be embedded into **neural network layers**
 - Can use **differentiable layers** in **PyTorch** and **TensorFlow 2.0**

```
from cvxpylayers.torch import CvxpyLayer
from cvxpylayers.tensorflow import CvxpyLayer
```



How does it work ?

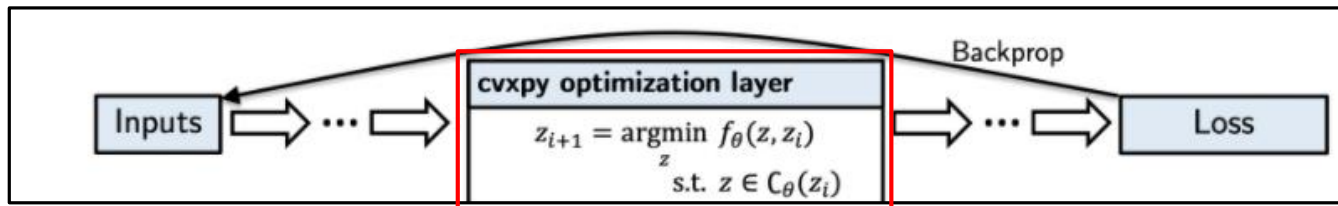
- Principles
 - Every convex program can be **canonicalized** as a cone program
 - Use derivative of a **conic solver**
 - Implicit function theorem
 - Optimality conditions (KKT conditions)
 - Apply **chain-rule** for canonicalization
- Pipeline



How can we use ?

- Usage

- All the users need is **high-level description** of problem



- Applications

- Finance (e.g., portfolio optimization)
- Engineering design (e.g., device sizing in electronic circuits)
- **Machine learning**, statistics (e.g., data fitting)
- Signal & image processing
- Control

Summary

- Contributions
 - Existing layers are difficult to use
 - Possible to differentiate convex programs with **high-level** language
 - Implement methodology in **CVXPY 1.1**
 - Implement **differentiable layers** for convex programs in **PyTorch** and **TensorFlow 2.0**
 - Improve efficiency (batched inputs, parallelization)
- Expected to be utilized in many research areas



Your Classifier is Secretly an Energy Based Model and You Should Treat it Like One

Jimin Seo

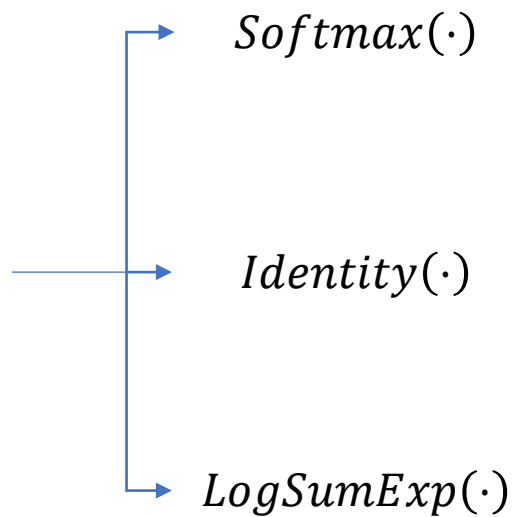
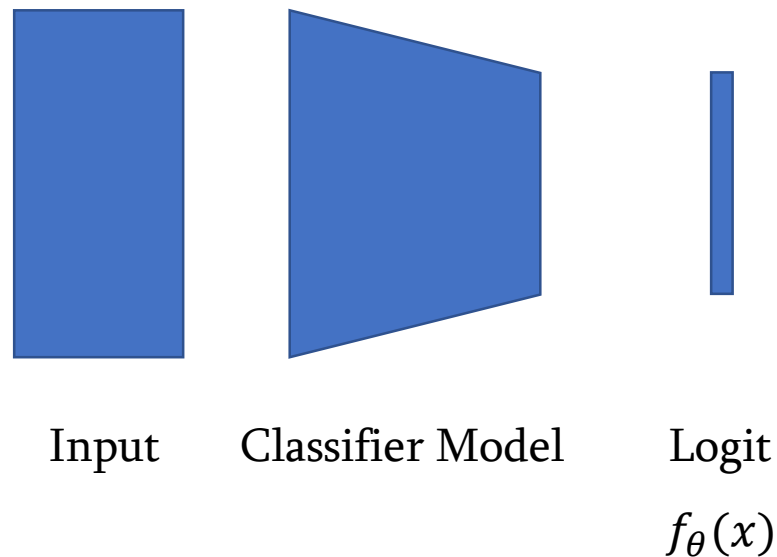
Introducing the Research

- Paper published on ICLR 2020
- Contributions are:
 1. Presented new framework of joint modeling of label and data
 2. Outperforms previous SoTA hybrid models at both generative and discriminative modeling
 3. Improvements on various tasks compared to hand-crafted methods

Main Idea

- Re-interpret the logit as probabilistic energy:

$$E_{\theta}(x) = -\text{LogSumExp}_y(f_{\theta}(x)[y]) = -\log \sum_y \exp(f_{\theta}(x)[y])$$



$$p_{\theta}(y|x) = \frac{\exp(f_{\theta}(x)[y])}{\sum_{y'} \exp(f_{\theta}(x)[y'])}$$

$$p_{\theta}(x, y) = \frac{\exp(f_{\theta}(x)[y])}{Z(\theta)}$$

$$p_{\theta}(x) = \frac{\sum_y \exp(f_{\theta}(x)[y])}{Z(\theta)}$$

Optimization

- We cannot compute/estimate $Z(\theta)$
 - Estimation of normalized density is intractable and MLE of θ is hard

$$E_{\theta}(x) = -\log \sum_y \exp(f_{\theta}(x)[y])$$

$$\frac{\partial}{\partial \theta} \log p_{\theta}(x) = \mathbb{E}_{p_{\theta}(x')} \left[\frac{\partial}{\partial \theta} E_{\theta}(x') \right] - \frac{\partial}{\partial \theta} E_{\theta}(x)$$



Stochastic Gradient Langevin Dynamics (SGLD) to draw sample from $p_{\theta}(x)$

Experiments

- Hybrid Modeling
- Calibration
- Out-of-Distribution Detection
- Robustness to adversarial examples

Experiments

- Hybrid Modeling

		Discriminative	Generative	
Class	Model	Accuracy% \uparrow	IS \uparrow	FID \downarrow
Hybrid	Residual Flow	70.3	3.6	46.4
	Glow	67.6	3.92	48.9
	IGEBM	49.1	8.3	37.9
	JEM $p(x y)$ factored	30.1	6.36	61.8
	JEM (Ours)	92.9	8.76	38.4
Disc.	Wide-Resnet	95.8	N/A	N/A
Gen.	SNGAN	N/A	8.59	25.5
	NCSN	N/A	8.91	25.32

Table 1: CIFAR10 Hybrid modeling Results. Residual Flow (Chen et al., 2019), Glow (Kingma & Dhariwal, 2018), IGEBM (Du & Mordatch, 2019), SNGAN (Miyato et al., 2018), NCSN (Song & Ermon, 2019)

Conclusion and Limitation

- Conclusion:
 - Proposed model of reinterpreted classifier architecture, combining strengths of discriminative and generative models
- Limitation:
 - Normalized likelihood cannot be computed
 - Gradient estimators are unstable when hyperparameter is not tuned

Paper Seminar

A Simple Framework for Contrastive Learning of Visual Representation

E In Son

Vehicle Intelligence Lab.

Prof. Seung-Woo Seo

November 19th, 2021

Paper Introduction

- **Title**

- A Simple Framework for Contrastive Learning of Visual Representations [1]

- **Author**

- T. Chen, S. Kornblith, M. Norouzi and G. Hinton
- Google Brain

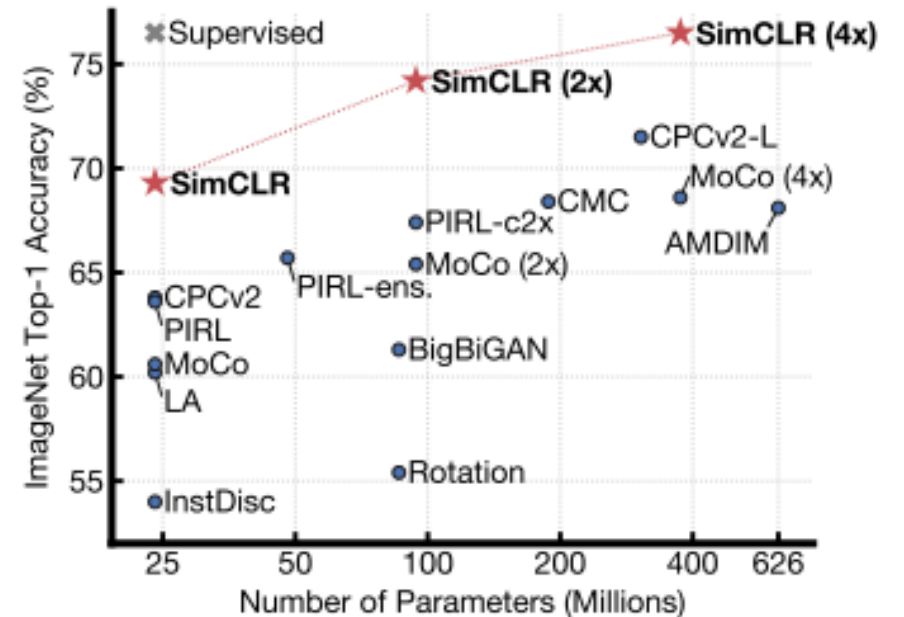
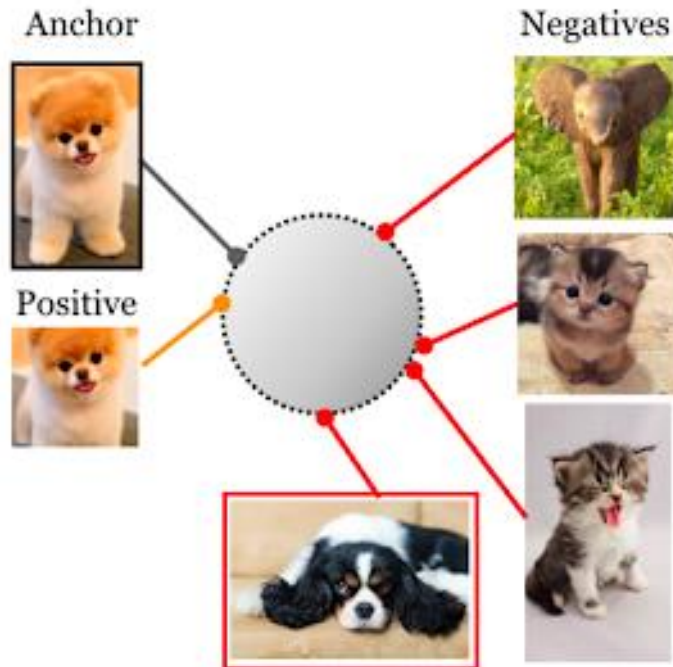
- **Publication**

- ICML 2020

SimCLR

▪ Main Idea

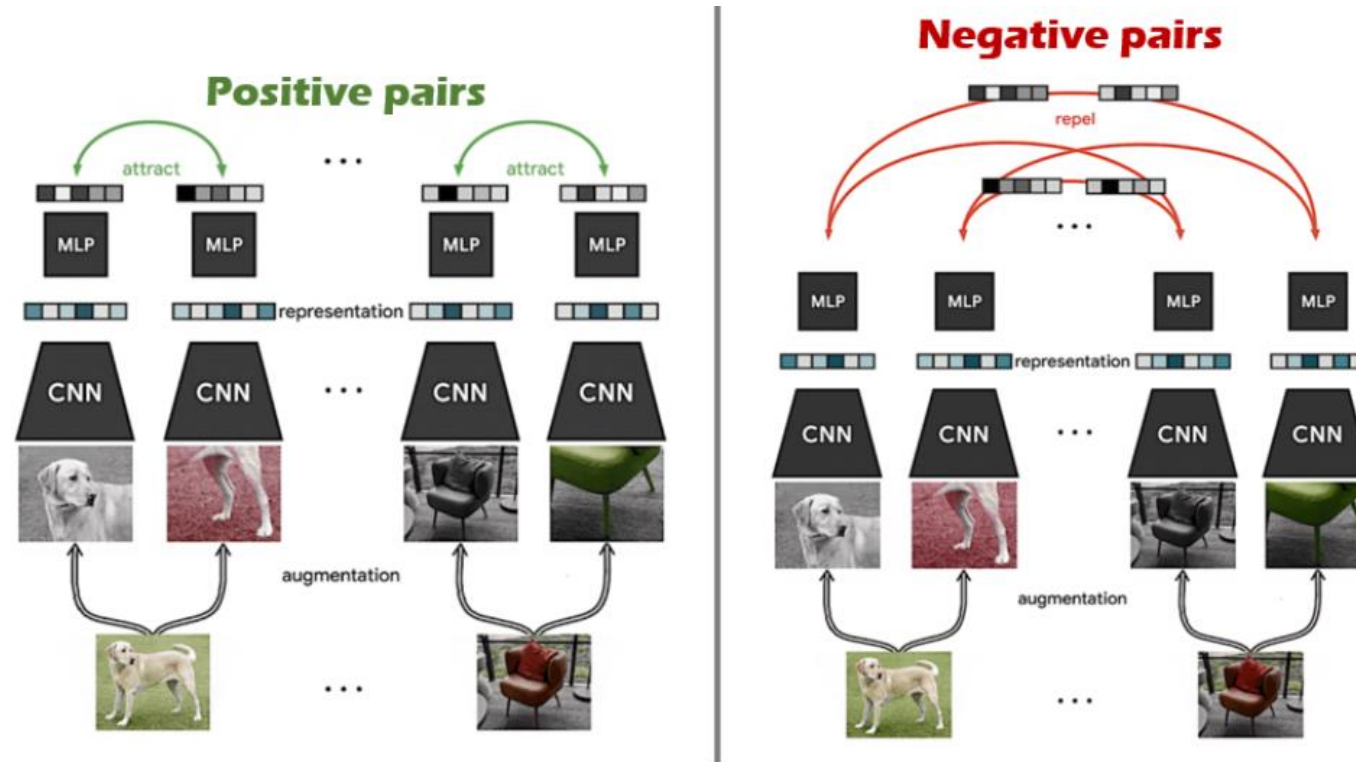
- contrastive learning (비슷한 Pair (positive pair)는 가깝게, 다른 Pair (negative pair)는 멀게 학습)을 이용하여 Visual Representation을 효과적으로 학습
- data augmentation을 이용하여 정답 label 없이 학습이 가능 → unsupervised learning



SimCLR

Architecture

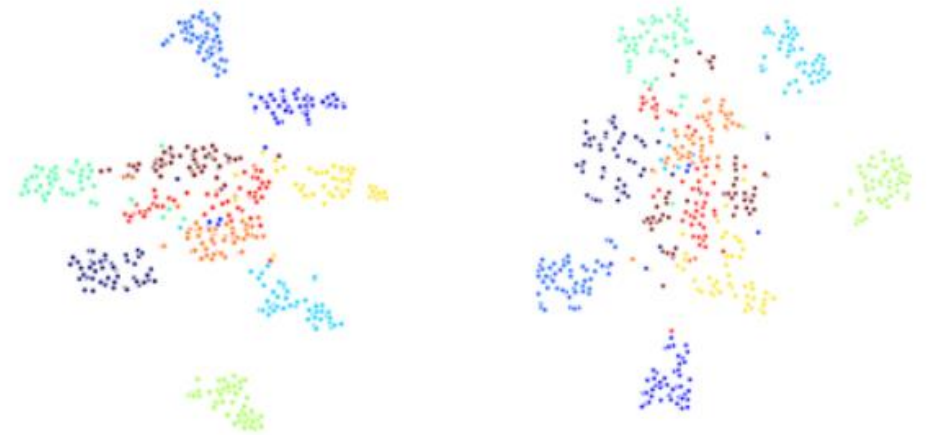
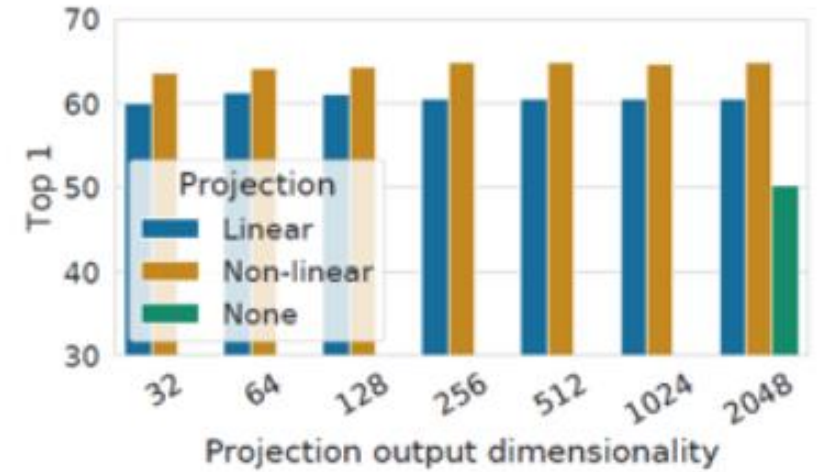
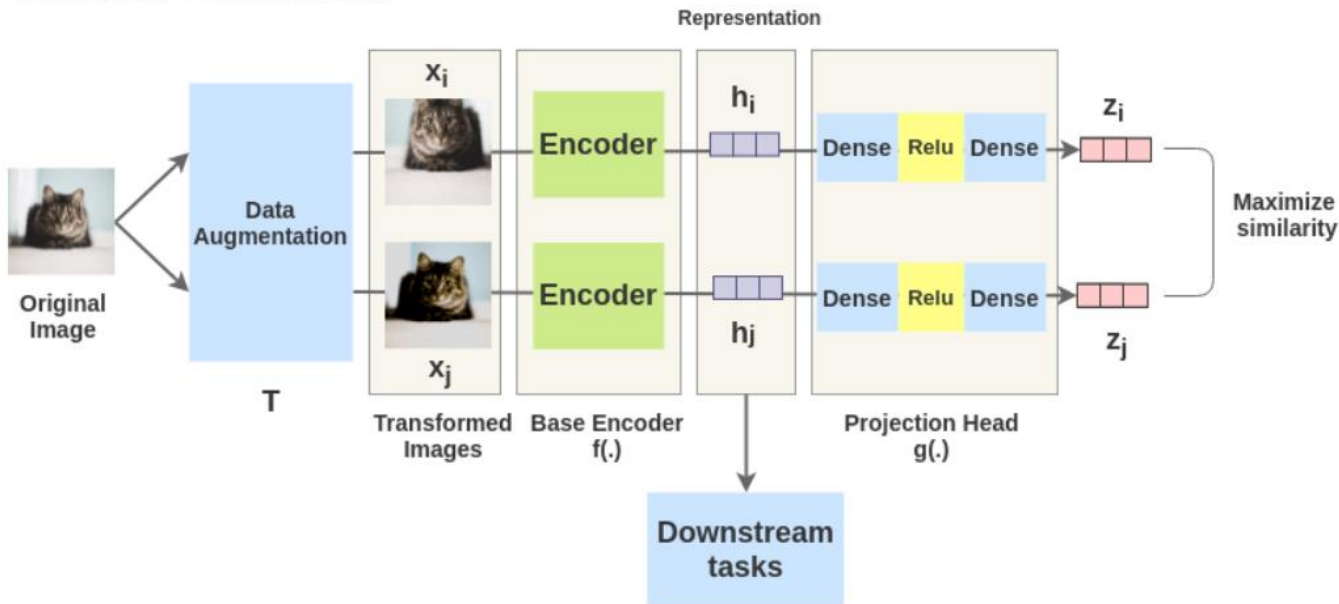
- for each training image, apply some data augmentation to create two different image
- minibatch of size N , we have $2N$ images. For each image, there are 1 positive and $2N-2$ negatives



SimCLR

- Training

SimCLR Framework



(a) h

(b) $z = g(h)$

SimCLR

Result

Method	Architecture	Param (M)	Top 1	Top 5
<i>Methods using ResNet-50:</i>				
Local Agg.	ResNet-50	24	60.2	-
MoCo	ResNet-50	24	60.6	-
PIRL	ResNet-50	24	63.6	-
CPC v2	ResNet-50	24	63.8	85.3
SimCLR (ours)	ResNet-50	24	69.3	89.0
<i>Methods using other architectures:</i>				
Rotation	RevNet-50 (4×)	86	55.4	-
BigBiGAN	RevNet-50 (4×)	86	61.3	81.9
AMDIM	Custom-ResNet	626	68.1	-
CMC	ResNet-50 (2×)	188	68.4	88.2
MoCo	ResNet-50 (4×)	375	68.6	-
CPC v2	ResNet-161 (*)	305	71.5	90.1
SimCLR (ours)	ResNet-50 (2×)	94	74.2	92.0
SimCLR (ours)	ResNet-50 (4×)	375	76.5	93.2

Method	Architecture	Label fraction	
		1%	10%
Top 5			
Supervised baseline	ResNet-50	48.4	80.4
<i>Methods using other label-propagation:</i>			
Pseudo-label	ResNet-50	51.6	82.4
VAT+Entropy Min.	ResNet-50	47.0	83.4
UDA (w. RandAug)	ResNet-50	-	88.5
FixMatch (w. RandAug)	ResNet-50	-	89.1
S4L (Rot+VAT+En. M.)	ResNet-50 (4×)	-	91.2
<i>Methods using representation learning only:</i>			
InstDisc	ResNet-50	39.2	77.4
BigBiGAN	RevNet-50 (4×)	55.2	78.8
PIRL	ResNet-50	57.2	83.8
CPC v2	ResNet-161(*)	77.9	91.2
SimCLR (ours)	ResNet-50	75.5	87.8
SimCLR (ours)	ResNet-50 (2×)	83.0	91.2
SimCLR (ours)	ResNet-50 (4×)	85.8	92.6

SimCLR

- **Result**

- fine-tuning

	Food	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
<i>Linear evaluation:</i>												
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
<i>Fine-tuned:</i>												
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

NORMFORMER: IMPROVED TRANSFORMER PRETRAINING WITH EXTRA NORMALIZATION

Sam Shleifer, Jason Weston, Myle Ott

2021-2 Pattern Recognition

2021.11.20

Seola Oh

Information Management Lab

Dept of Industrial Engineering

Seoul National University



1. Introduction

- original transformer architecture (Vaswani et al., 2017)¹ applies Layer Normalization after each sublayer's residual connection ("Post-LN")

$$\text{PostLN}(x) = \text{LayerNorm}(x + \text{Sublayer}(x))$$

$$\text{LayerNorm}(x) = \frac{x - E[x]}{\sqrt{\text{Var}[x] + \epsilon}} \cdot \gamma + \beta$$

- Post-LN transformers tend to have larger magnitude gradients in later layers compared to earlier layers
- moving the LayerNorm operation to the beginning of each sublayer ("Pre-LN") (Xiong et al., 2020)²

$$\text{PreLN}(x) = x + \text{Sublayer}(\text{LayerNorm}(x)).$$

- improves stability over Post-LN
- but gradients at earlier layers tend to be larger than gradients at later layers
- propose NormFormer

1) Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, 2017.

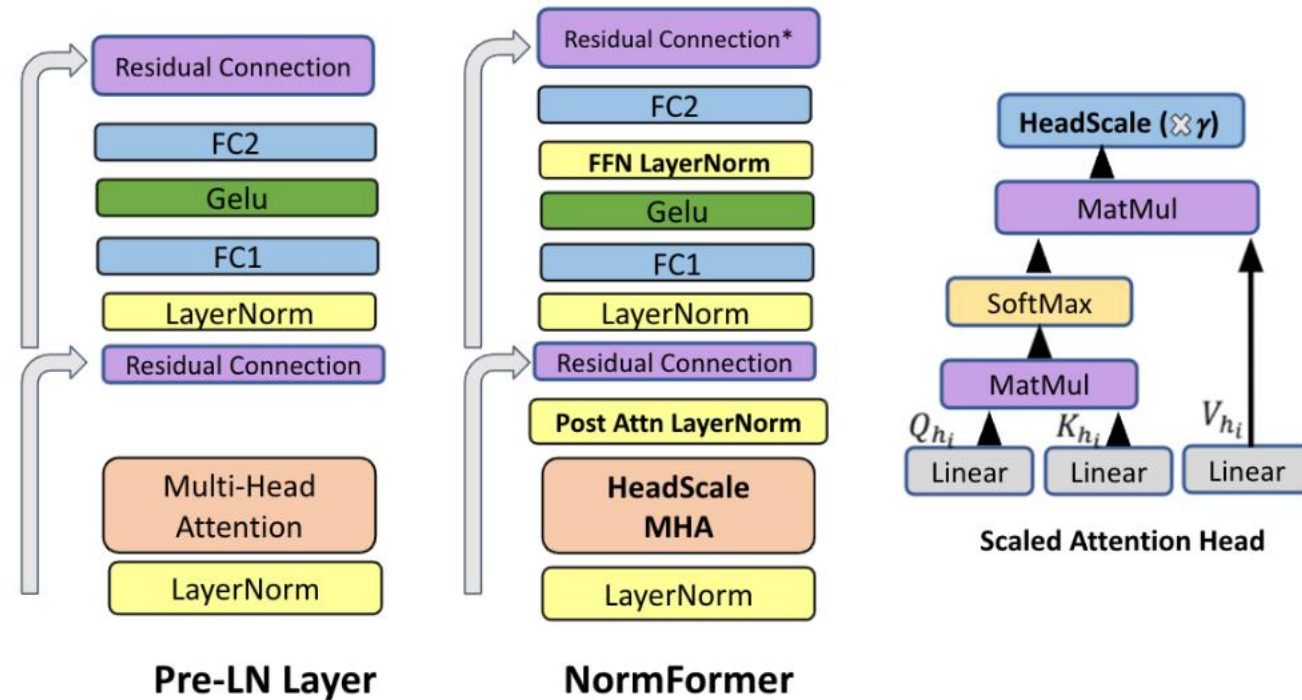
2) Baolin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zhang, Chen Yang, Hishui Zhang, Yanyun Luo, Liwei Wang, and Tie Yan Liu. On layer normalization in the transformer architecture, 2020.



alleviates the gradient magnitude mismatch by adding 3 normalization operations to each layer

1. Introduction

- propose NormFormer
 - alleviates the gradient magnitude mismatch by adding 3 normalization operations to each layer



2. Approach

2.1 NormFormer

1) Scaling Attention Heads

- head-wise scaling inside the attention module
- standard multi-head attention operation

$$\text{MultiHeadAttention}(Q, K, V) = \text{Concat}(\mathbf{h}_1, \dots, \mathbf{h}_n)W^O$$

$$\mathbf{h}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V,$$

- propose scaling the output of each attention head via learned scalar coefficients γ_i

$$\text{HeadScaleMHA}(Q, K, V) = \text{Concat}(\gamma_1\mathbf{h}_1, \dots, \gamma_n\mathbf{h}_n)W^O$$



2. Approach

2) Additional Layer Normalization and Putting it All Together

- add LayerNorm operation after the attention module
- add LayerNorm operation after the first fully connected layer
- in the Pre-LN transformer each layer modifies an input as follows

$$x_{l+1}^{\text{PreLN}} = \text{FFN}(\text{MHA}(x_l))$$

where $\text{MHA}(x) = x + \text{MultiHeadAttention}(\text{LN}(x), \text{LN}(x), \text{LN}(x))$

$$\text{FFN}(x) = x + \sigma(\text{LN}(x)W_1 + b_1)W_2 + b_2$$

$$\text{LN}(x) = \text{LayerNorm}(x)$$

- NormFormer instead modifies each input as

$$x_{l+1}^{\text{NormFormer}} = \text{NormFFN}(\text{NormScaledMHA}(x_l))$$

where $\text{NormScaledMHA}(x) = x + \mathbf{LN}(\mathbf{HeadScaleMHA}(\text{LN}(x), \text{LN}(x), \text{LN}(x)))$

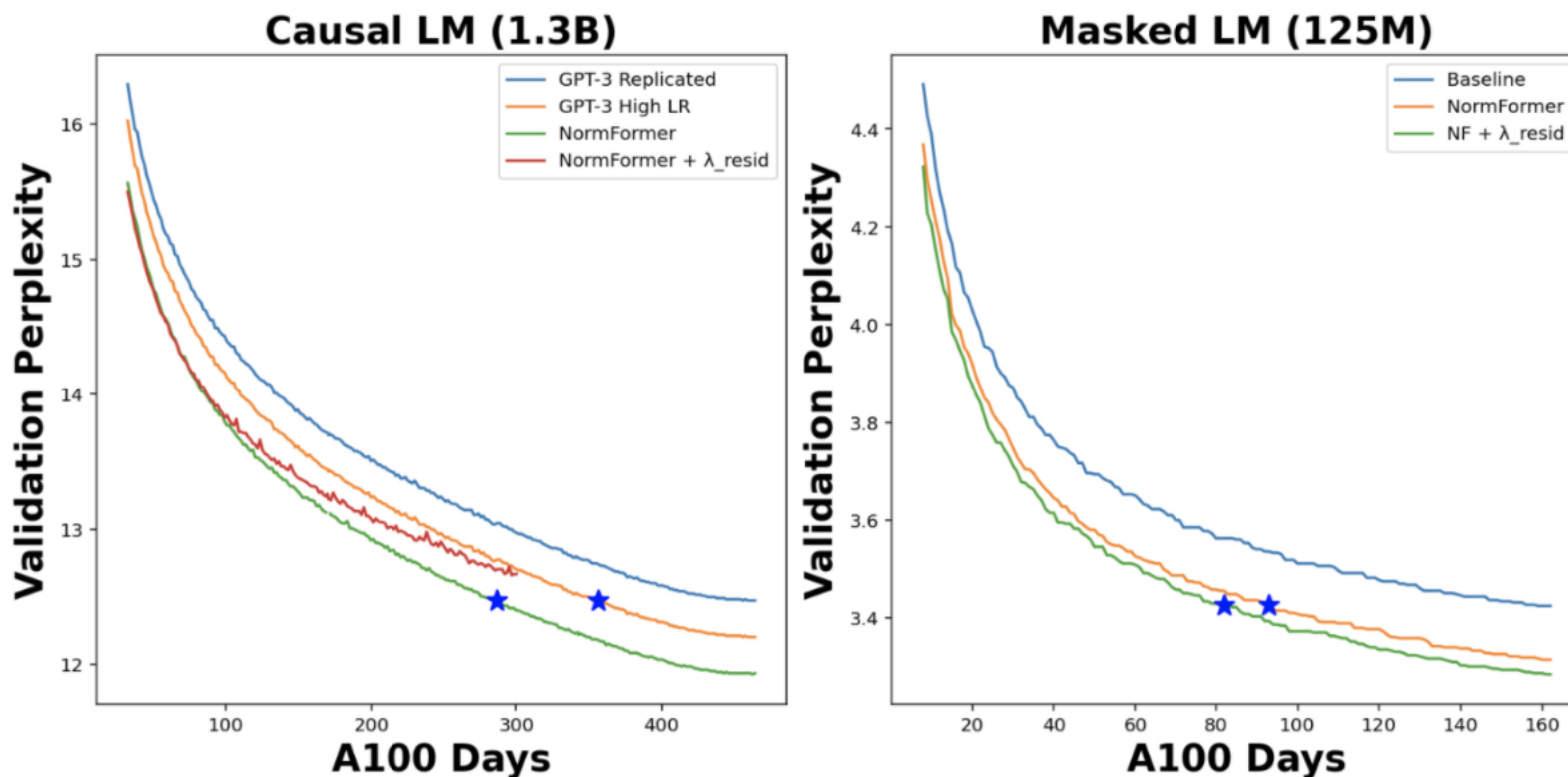
$$\text{NormFFN}(x) = \mathbf{x} + \mathbf{LN}(\sigma(\text{LN}(x)W_1 + b_1))W_2 + b_2$$

- introduce a small number of additional learnable parameters



3. Results

- report pretraining perplexities for CLMs and MLMs as a function of training wall-time
- NormFormer trains significantly faster and achieves better validation perplexities for a given training compute budget



3. Results

- similar trend on downstream tasks

1) zero shot accuracy for causal LMs using the tasks and prompt from Brown et al. (2020)

- NormFormer outperforms GPT-3 at all sizes

	$ \theta $	LR	$Relu^2$	λ_{resid}	Steps	PPL	HS	PI	WG	SC	OB	Avg
Random Baseline	-	-	-	-	-	-	25.0	50.0	50.0	50.0	25.0	40.0
GPT3-125M (paper)	124.4	6e-4	-	-	572K	-	33.7	64.6	52.0	63.3	35.6	49.8
GPT3-125M (replicated)	124.4	6e-4	-	-	572K	21.11	33.7	66.5	52.2	66.1	35.4	50.8
GPT3-125M (High LR)	124.4	3e-3	-	-	572K	21.09	35.3	67.5	50.5	66.3	35.0	50.9
NormFormer-125M	124.5	3e-3	-	-	540K	20.34	34.9	67.1	52.3	66.3	38.0	51.7
NormFormer-125M	124.5	3e-3	-	✓	539K	20.11	34.9	65.9	53.4	67.5	40.0	52.3
GPT3-355M (paper)	354.7	3e-4	-	-	572K	-	43.6	70.2	52.1	68.5	43.2	55.5
GPT3-355M (replicated)	354.7	3e-4	-	-	572K	15.41	46.1	70.8	54.6	71.1	41.2	56.8
GPT3-355M (High LR)	354.7	1e-3	-	-	572K	14.85	48.4	71.7	53.8	73.3	43.4	58.1
NormFormer-355M	355.0	1e-3	-	-	552K	14.54	49.7	71.8	56.0	73.8	43.6	59.0
NormFormer-355M	355.0	1e-3	-	✓	550K	14.52	49.7	72.0	56.7	73.2	43.8	59.1
GPT3-1.3B (paper)	1313.5	2e-4	-	-	286K	-	54.7	75.1	58.0	73.4	46.8	61.6
GPT3-1.3B (replicated)	1313.5	2e-4	-	-	286K	12.56	58.5	74.6	58.1	76.8	49.4	63.5
GPT3-1.3B (High LR)	1313.5	6e-4	-	-	286K	12.21	57.5	74.3	59.3	76.3	50.8	63.6
NormFormer-1.3B	1314.0	6e-4	-	-	275K	11.94	60.5	74.5	60.1	77.5	50.8	64.7
GPT3-2.7B (paper)	2648.7	1.6e-4	-	-	286K	-	62.8	75.6	62.3	77.2	53.0	66.2
GPT3-2.7B (replicated)	2648.7	1.6e-4	-	-	286K	10.92	65.9	76.6	61.4	78.2	49.6	66.3
NormFormer-2.7B	2649.5	6e-4	✓	-	277K	10.55	68.1	78.1	64.4	79.4	53.4	68.7
GPT3-2.7B-Relu	2648.7	1.6e-4	✓	-	230K	10.99	65.9	76.1	63.2	79.3	49.4	66.8
GPT3-2.7B-Relu	2648.7	6e-4	✓	-	28K	-	diverged					-
NormFormer-2.7B	2649.5	6e-4	✓	-	222K	10.73	67.4	77.2	64.4	78.9	52.6	68.1

2) for MLM models, fine-tuned accuracy on GLUE

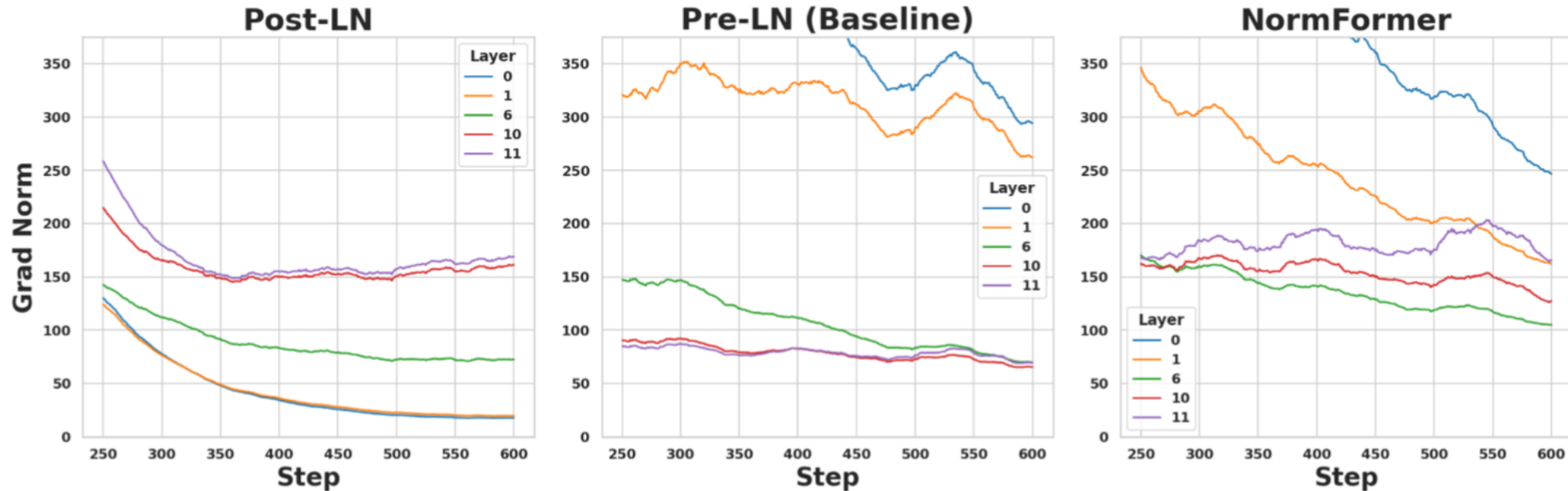
- NormFormer MLM models outperform their Pre-LN counterparts on every task

	Model Size	λ_{resid}	PPL	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	Avg
Baseline	125.42	-	3.42	74.3	85.9	84.6	91.6	90.7	66.4	92.9	83.77
NormFormer	125.50	-	3.31	82.6	86.3	86.0	91.9	91.3	67.9	93.8	85.69
NormFormer	125.51	✓	3.29	80.9	86.2	85.3	91.5	91.2	62.8	94.2	84.59



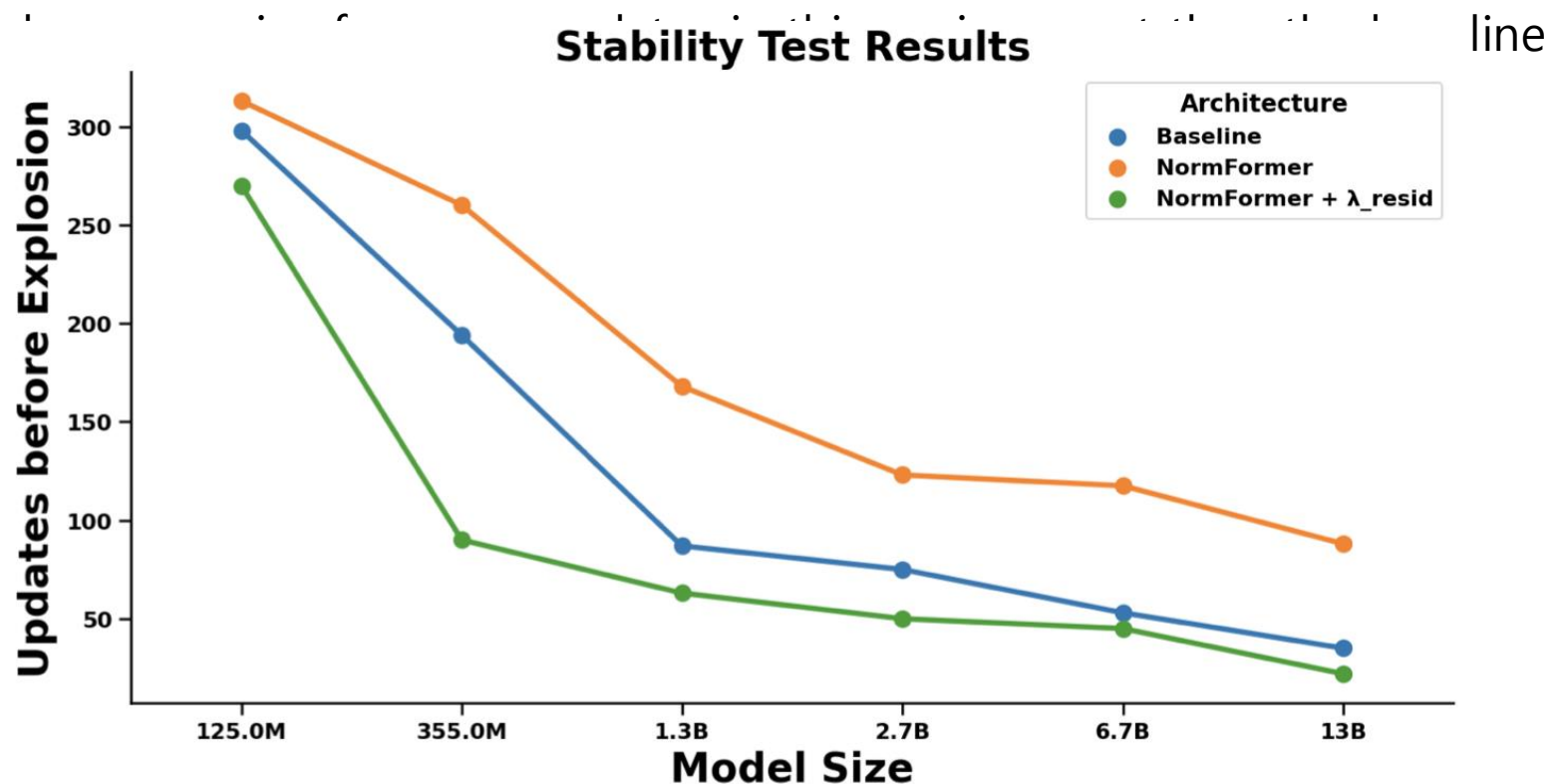
4. Analysis

- average L1 norm of the gradients to the second fully connected weight in various layers for a 12 layer, 12.5M parameter CLM model at the beginning of training
- NormFormer brings the average gradient norms closer together for different layers in the network



4. Analysis

- One result of reducing the gradient mismatch is the ability to train stably with larger learning rates
- train it on a learning rate schedule with a very large peak learning rate, so that the learning rate increases a little each step until the loss explodes
- NormFormer mod



5. Ablations

- removing any of our additions to the transformer block degrades performance on language modeling tasks

Architecture	Valid PPL
NormFormer+ResScale	15.88
- Post-Attn LN	15.92
- FFN LN	16.14
- Head Scale	16.22
- Res Scale	16.20
+ 3 More LN	15.88
Baseline	16.37



6. Conclusion

- mismatch in the gradients of Pre-LN transformer weights
 - earlier layers receive much larger gradients than later layers
- propose NormFormer, which alleviates these issues by adding 3 extra operations to each transformer layer
 - help the gradient mismatch for fully connected parameters
 - improve validation perplexity
 - downstream task performance for both causal and masked language models
- None can be removed without degrading performance back towards the baseline
- adding more normalization does not improve performance



Thank you





Deblurring using Analysis-Synthesis Networks Pair

CVPR 2020 paper

ISPL 오영진



Analysis Network

- $B = I * k$
 - If we know k , it is non-blind deblurring
 - If we do not know k , it is blind deblurring
- Recover the blur-kernel k (estimated)
- Utilizes a novel ‘cross-correlation layer’
- Feeds the output kernel to the synthesis network to make the problem ‘non-blind’

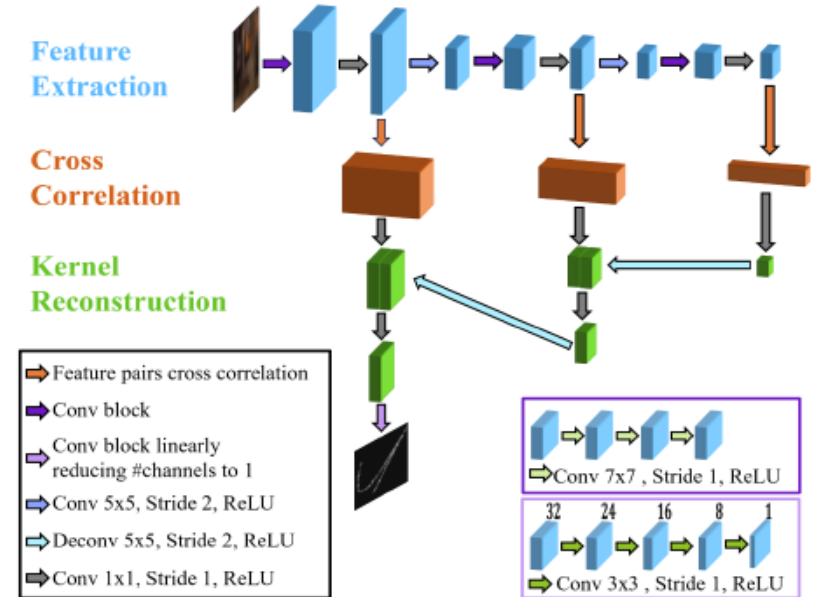


Figure 2. Analysis Network Architecture. The first stage consists of extracting features (activations) at multiple scales by applying convolution layers, and pooling operations. At the second stage, the cross-correlation between the resulting activations are computed at all scales. Finally, the estimated blur-kernel is reconstructed from coarse to fine by means of un-pooling and convolution steps. The pooling and up-pooling operations apply x2 scaling. We use 64 filters in the feature extracting step, and reduce them to 32 channels before computing the cross-correlation stage. This illustration shows only two spatial scalings, whereas our implementation consists of three.



Cross-correlation layer

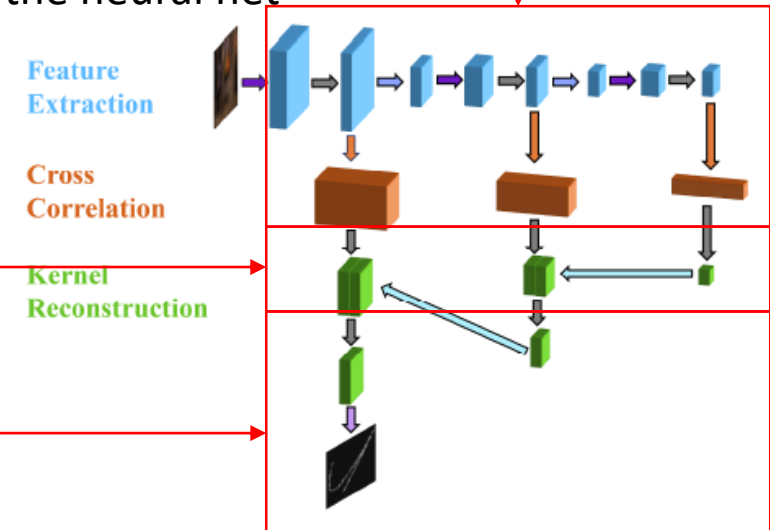
- Auto-correlation of blurry images can be utilized to estimate blur-kernels
-> incorporate this knowledge into layers of the neural network

1) Compute cross-correlation between activation maps

$C_{i,j}(s, t) = \sum_{x,y} f_i(x - s, y - t) f_i(x, y)$ at activation maps f_i at spatial range $-2^{-l}m \leq s, t \leq 2^{-l}m$, where m is the spatial dimension of the kernel grid, and l is the scale level of the neural net

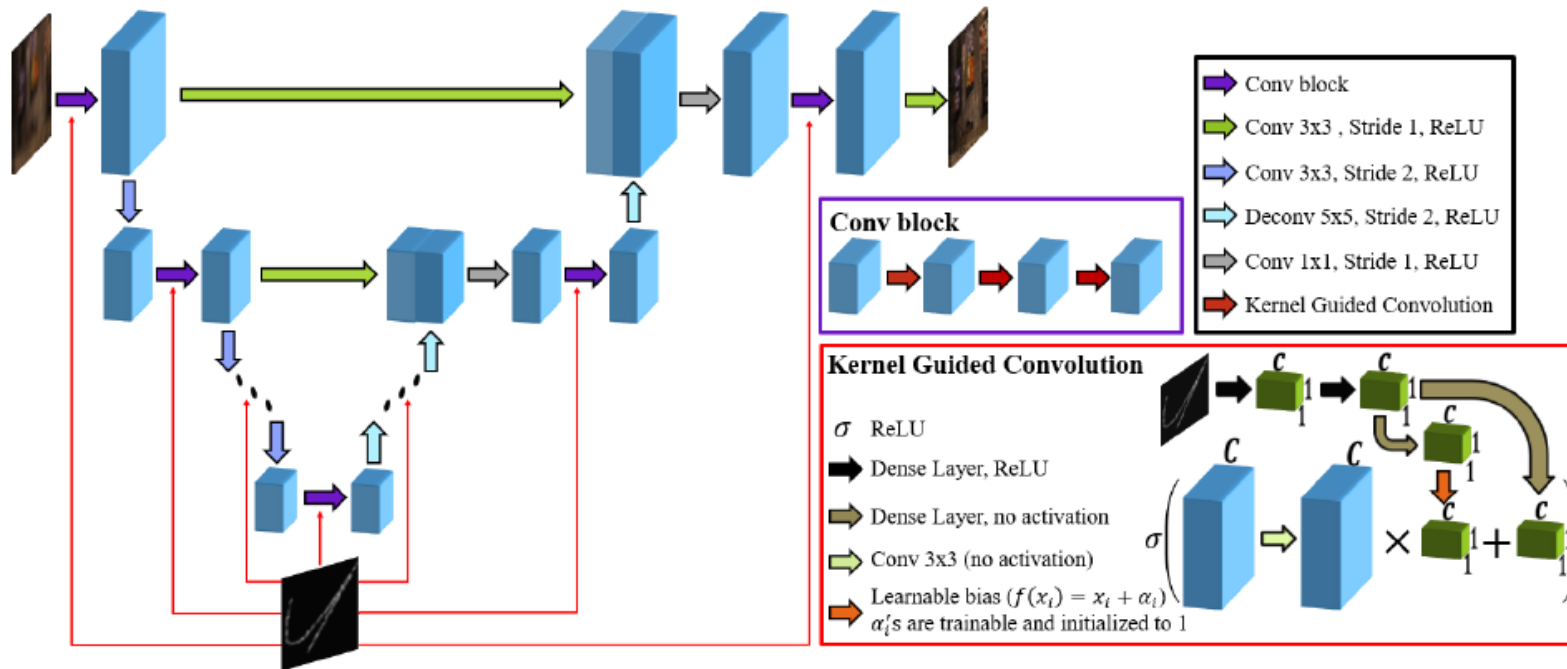
2) Reduce channel size to 32

3) Concatenated and integrated to finest scale



Synthesis Network

Kernel Guided Convolution $r = r \odot (1 + m(k)) + b(k)$, incorporates the kernel information in the synthesis network as a **prior** and dictates the network's actions



Configuration	PSNR
Synthesis + GT Kernels	
No guidance	24.80
Additive guidance	28.58
Multiplicative guidance	28.41
Additive+Multiplicative guidance	28.73

Figure 3. Synthesis Network Architecture. A standard U-Net is augmented with kernel guided convolutions at all its layers. As shown in the red schematic, the activations in these convolutions are modulated and biased by vectors derived from the blur-kernel using FC layers. Each convolution layer consists of 3 successive convolutions using 128 filters, separated by ReLU activations.



Decision Transformer: Reinforcement Learning via Sequence Modeling

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, Igor Mordatch

NeurIPS 2021

오우석



DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
SEOUL NATIONAL UNIVERSITY

RILAB
<http://rllab.snu.ac.kr>

- Bootstrapping problem

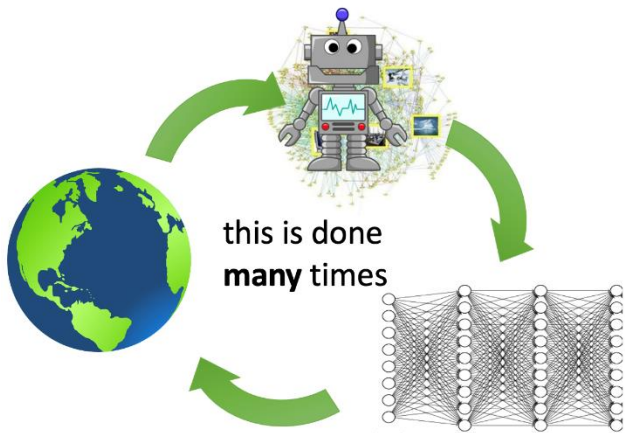
$$Q(s, a) \leftarrow Q(s, a) + \alpha(R_{t+1} + \gamma Q(s', a') - Q(s, a))$$

- Credit assignment problem

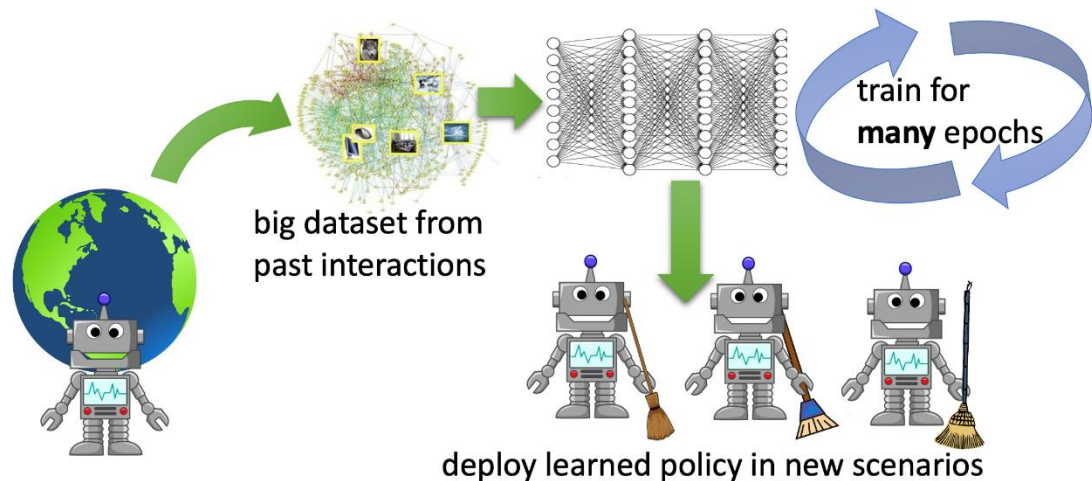


- Train in collected dataset
- Unlike other RLs, it does not interact (high-cost) with the environment.
- because of the distribution shift between policies, it is difficult to train

reinforcement learning

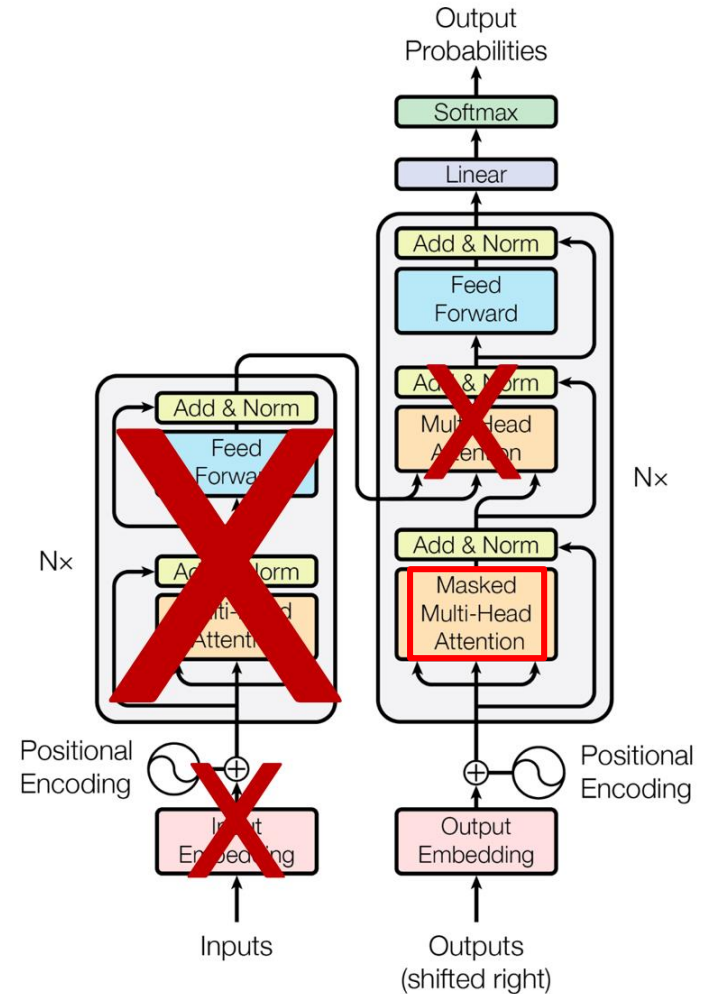
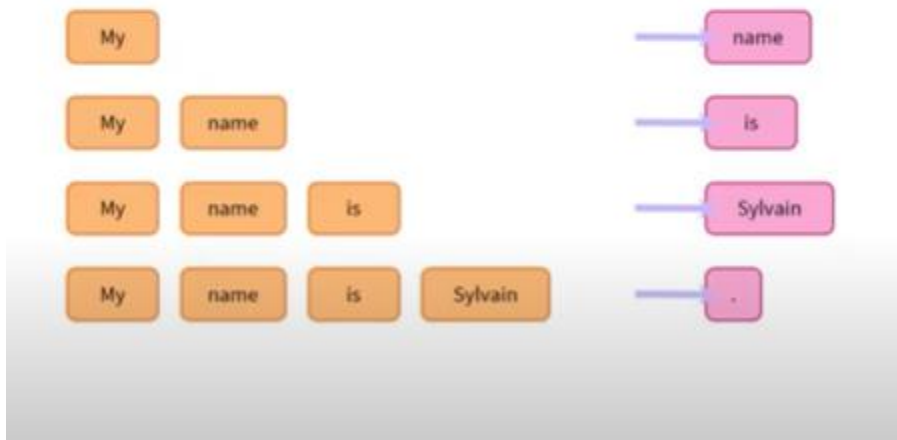


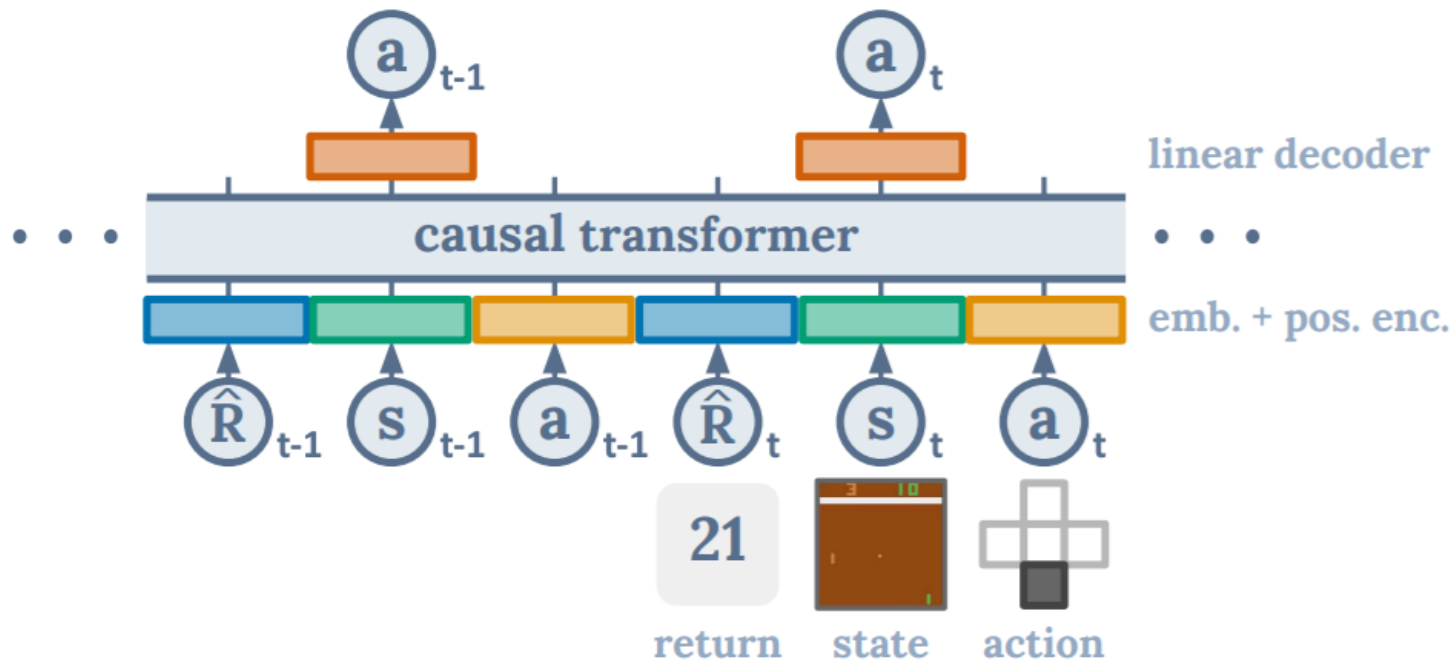
offline reinforcement learning



Transformer – GPT model

Causal Language Modeling Guessing the next word in a sentence





s_t : state of time t

r_t : reward of time t

a_t : action of time t

$$\hat{R}_t = \sum_{t'=t}^T r_{t'} : \text{returns-to-go}$$



```
# main model
def DecisionTransformer(R, s, a, t):
    # compute embeddings for tokens
    pos_embedding = embed_t(t) # per-timestep (note: not per-token)
    s_embedding = embed_s(s) + pos_embedding
    a_embedding = embed_a(a) + pos_embedding
    R_embedding = embed_R(R) + pos_embedding

    # interleave tokens as (R_1, s_1, a_1, ..., R_K, s_K)
    input_embs = stack(R_embedding, s_embedding, a_embedding)

    # use transformer to get hidden states
    hidden_states = transformer(input_embs=input_embs)

    # select hidden states for action prediction tokens
    a_hidden = unstack(hidden_states).actions

    # predict action
    return pred_a(a_hidden)

# training loop
for (R, s, a, t) in dataloader: # dims: (batch_size, K, dim)
    a_preds = DecisionTransformer(R, s, a, t)
    loss = mean((a_preds - a)**2) # L2 loss for continuous actions
    optimizer.zero_grad(); loss.backward(); optimizer.step()
```



```
# main model
def DecisionTransformer(R, s, a, t):
    # compute embeddings for tokens
    pos_embedding = embed_t(t) # per-timestep (note: not per-token)
    s_embedding = embed_s(s) + pos_embedding
    a_embedding = embed_a(a) + pos_embedding
    R_embedding = embed_R(R) + pos_embedding

    # interleave tokens as (R_1, s_1, a_1, ..., R_K, s_K)
    input_embeds = stack(R_embedding, s_embedding, a_embedding)

    # use transformer to get hidden states
    hidden_states = transformer(input_embeds=input_embeds)

    # select hidden states for action prediction tokens
    a_hidden = unstack(hidden_states).actions

    # predict action
    return pred_a(a_hidden)

# evaluation loop
target_return = 1 # for instance, expert-level return
R, s, a, t, done = [target_return], [env.reset()], [], [1], False
while not done: # autoregressive generation/sampling
    # sample next action
    action = DecisionTransformer(R, s, a, t)[-1] # for cts actions
    new_s, r, done, _ = env.step(action)

    # append new tokens to sequence
    R = R + [R[-1] - r] # decrement returns-to-go with reward
    s, a, t = s + [new_s], a + [action], t + [len(R)]
    R, s, a, t = R[-K:], ... # only keep context length of K
```



- It showed good performance in most datasets and environments(SOTA in offline RL).
- This method performed better than other methods in key-to-door problems where long-term credit assignment is important.



RLLAB

<http://rllab.snu.ac.kr>



Learning Continuous Image Representation with Local Implicit Image Function

Yinbo Chen, Sifei Liu, Xiaolong Wang

CVPR 2021

Super-resolution

- Producing high-resolution(HR) images from a corresponding low-resolution(LR) image.
 - Ill-posed problem



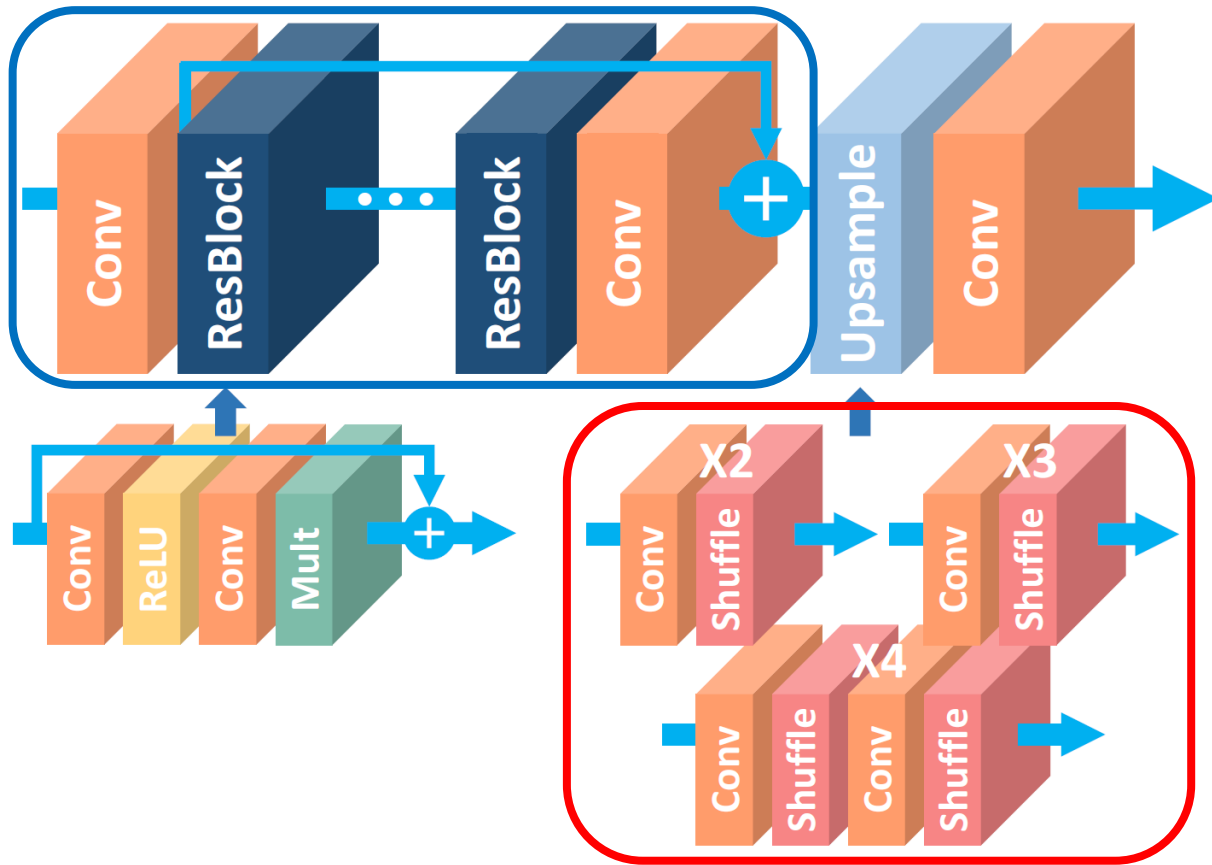
Low-resolution image



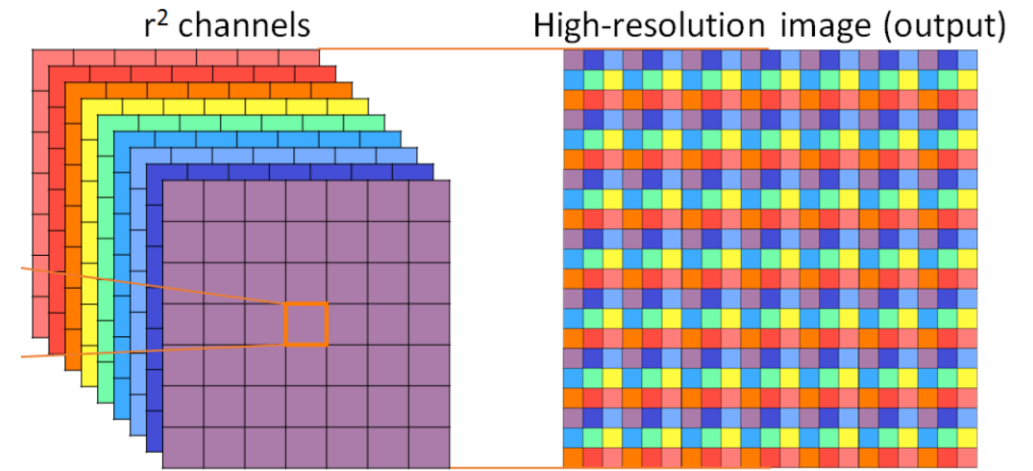
High-resolution image

Related Work

- EDSR (Lim et al., 2017)



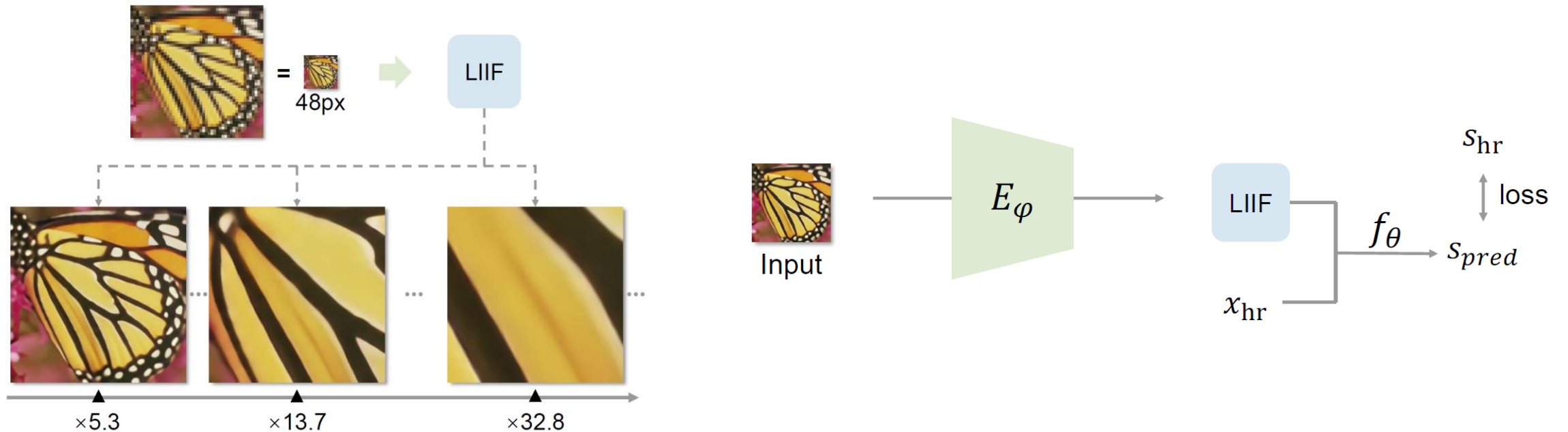
Lim et al., 2017



Shi et al., 2016

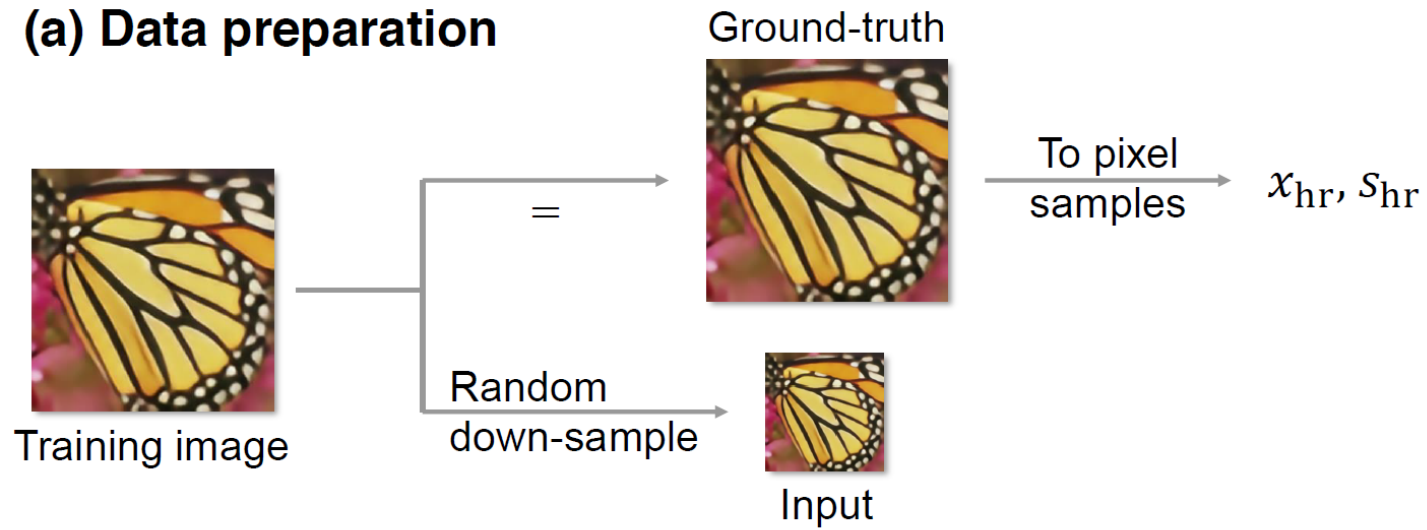
Summary

- Super-resolution method for arbitrary resolution
 - Previous methods had fixed high resolution for each model. (x2,x3,x4,...)
- Takes an image coordinate and surrounding deep features to predict the pixel at a given coordinate.

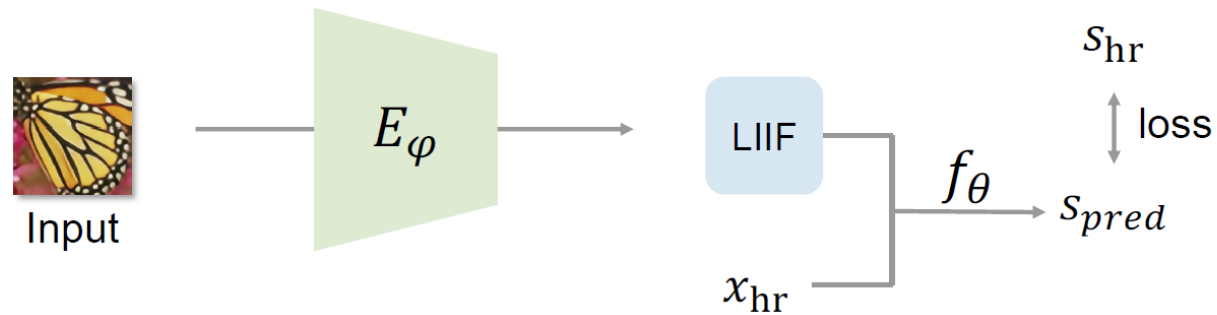


Method

(a) Data preparation

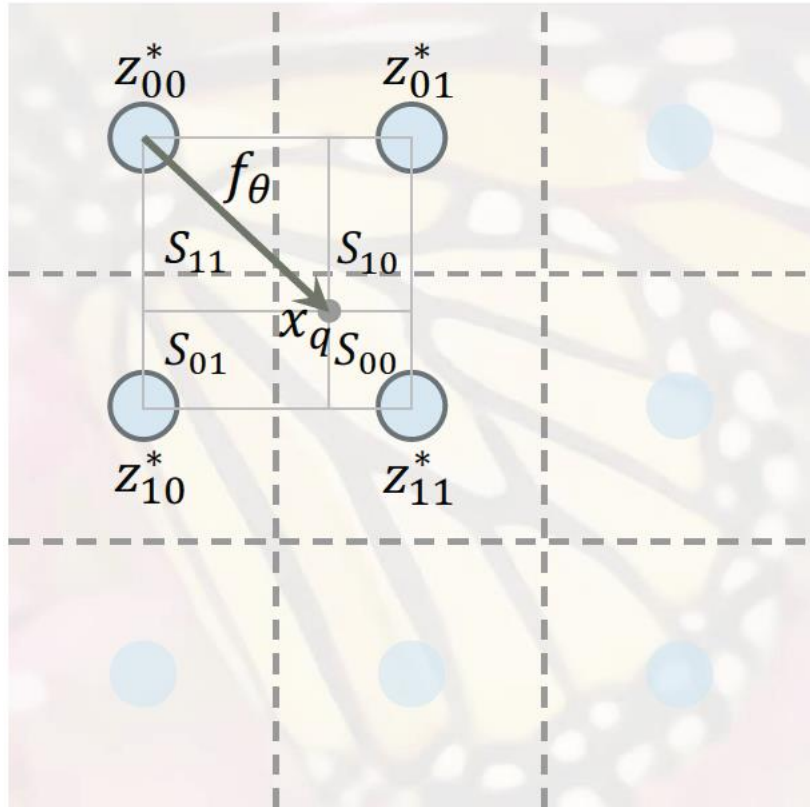


(b) Training

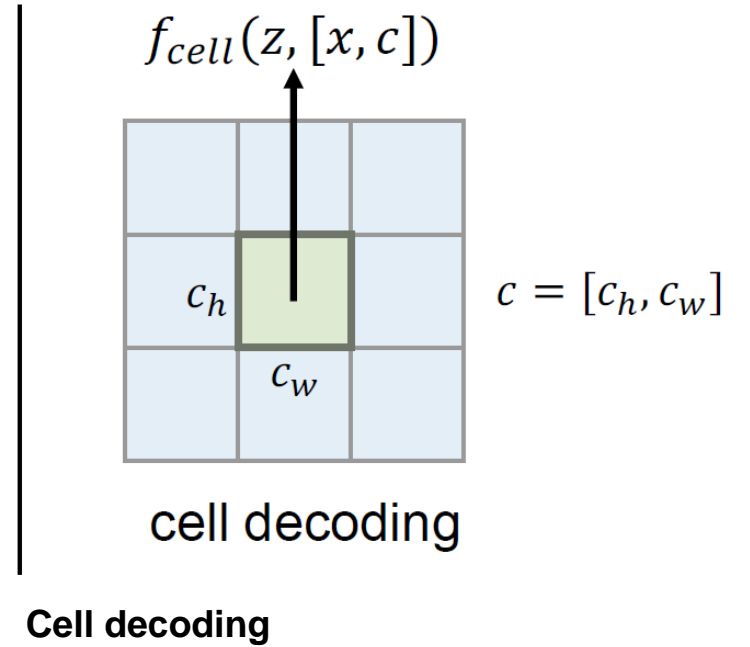
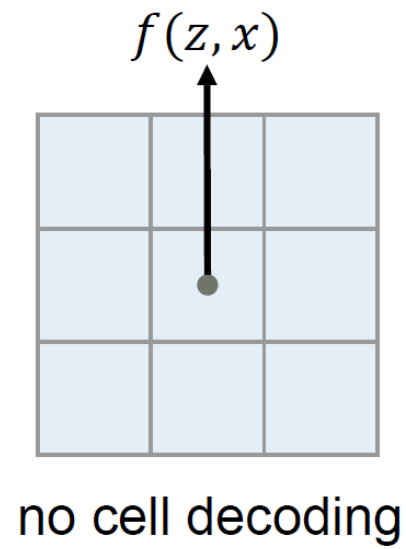


$$I^{(i)}(x_q) = f_\theta(z^*, x_q - v^*)$$

Method



LIF representation with local ensemble



Results

Results on benchmark datasets (PSNR(dB))

Dataset	Method	In-distribution			Out-of-distribution	
		$\times 2$	$\times 3$	$\times 4$	$\times 6$	$\times 8$
Set5	RDN [51]	38.24	34.71	32.47	-	-
	RDN-MetaSR [#] [15]	38.22	34.63	32.38	29.04	26.96
	RDN-LIIF (ours)	38.17	34.68	32.50	29.15	27.14
Set14	RDN [51]	34.01	30.57	28.81	-	-
	RDN-MetaSR [#] [15]	33.98	30.54	28.78	26.51	24.97
	RDN-LIIF (ours)	33.97	30.53	28.80	26.64	25.15
B100	RDN [51]	32.34	29.26	27.72	-	-
	RDN-MetaSR [#] [15]	32.33	29.26	27.71	25.90	24.83
	RDN-LIIF (ours)	32.32	29.26	27.74	25.98	24.91
Urban100	RDN [51]	32.89	28.80	26.61	-	-
	RDN-MetaSR [#] [15]	32.92	28.82	26.55	23.99	22.59
	RDN-LIIF (ours)	32.87	28.82	26.68	24.20	22.79

Conclusion

- LIIF learns continuous representation in 2D image.
- LIIF representation can produce high-resolution images in arbitrary scale with high fidelity.

패턴인식 기말 과제

EfficientNetV2: Smaller Models and Faster Training

유승찬

2021-11-17



Seoul National University
Communications & Machine Learning Lab.

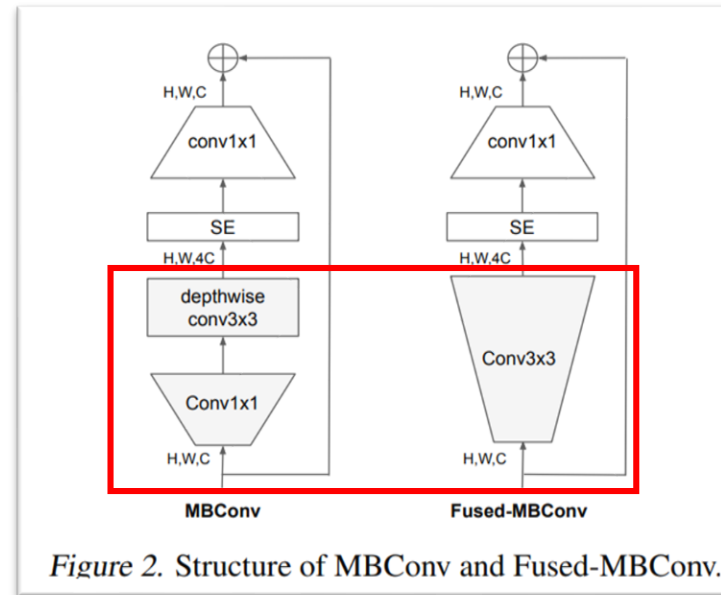
EfficientNet (Version 1)

- Tan, Mingxing, et al. "EfficientNet: Rethinking Model Scaling for convolutional Neural Networks." *International conference on machine learning*. PMLR, 2019
- <https://arxiv.org/pdf/1905.11946.pdf>
- "Compound Scaling"

$$\begin{aligned} \text{depth: } d &= \alpha^\phi \\ \text{width: } w &= \beta^\phi \\ \text{resolution: } r &= \gamma^\phi \end{aligned} \tag{3}$$
$$\begin{aligned} \text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 &\approx 2 \\ \alpha \geq 1, \beta \geq 1, \gamma &\geq 1 \end{aligned}$$

Main idea of EfficientNetV2

- Progressive Learning : Main Topic, to be discussed.
- Fused MBConv.



- Non-uniform Scaling : Slightly modify the compound scaling rule at later stages

Progressive Learning

- The accuracy drop comes from the unbalanced regularization
- When training with different image sizes, we should also adjust the regularization strength accordingly

Table 5. ImageNet top-1 accuracy. We use RandAug (Cubuk et al., 2020), and report mean and stdev for 3 runs.

	Size=128	Size=192	Size=300
RandAug magnitude=5	78.3 ±0.16	81.2 ±0.06	82.5 ±0.05
RandAug magnitude=10	78.0 ±0.08	81.6 ±0.08	82.7 ±0.08
RandAug magnitude=15	77.7 ±0.15	81.5 ±0.05	83.2 ±0.09

Adaptive Regularization



Bigger Image size



Strong Regularization
(RandAugment, Dropout, Mixup)

Algorithm 1 Progressive learning with adaptive regularization.

Input: Initial image size S_0 and regularization $\{\phi_0^k\}$.

Input: Final image size S_e and regularization $\{\phi_e^k\}$.

Input: Number of total training steps N and stages M .

for $i = 0$ **to** $M - 1$ **do**

Image size: $S_i \leftarrow S_0 + (S_e - S_0) \cdot \frac{i}{M-1}$

Regularization: $R_i \leftarrow \{\phi_i^k = \phi_0^k + (\phi_e^k - \phi_0^k) \cdot \frac{i}{M-1}\}$

Train the model for $\frac{N}{M}$ steps with S_i and R_i .

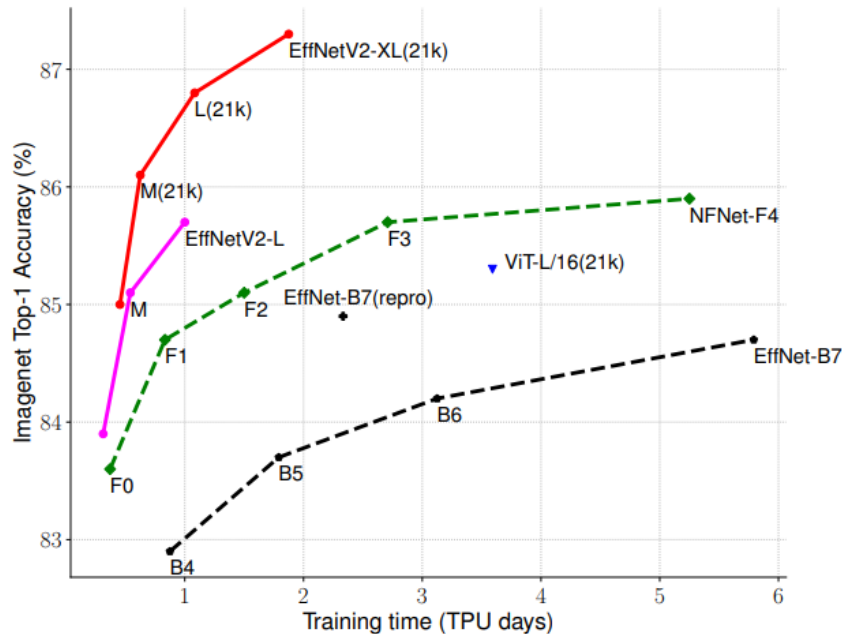
end for

Table 6. Progressive training settings for EfficientNetV2.

	S		M		L	
	min	max	min	max	min	max
Image Size	128	300	128	380	128	380
RandAugment	5	15	5	20	5	25
Mixup alpha	0	0	0	0.2	0	0.4
Dropout rate	0.1	0.3	0.1	0.4	0.1	0.5

Results

- Better Training/Parameter efficiency



(a) Training efficiency.

	EfficientNet (2019)	ResNet-RS (2021)	DeiT/ViT (2021)	EfficientNetV2 (ours)
Top-1 Acc.	84.3%	84.0%	83.1%	83.9%
Parameters	43M	164M	86M	24M

(b) Parameter efficiency.

- Progressive learning for other networks

Table 12. Progressive learning for ResNets and EfficientNets – (224) and (380) denote inference image size. Our progressive training improves both accuracy and training time for all networks.

	Baseline		Progressive	
	Acc.(%)	TrainTime	Acc.(%)	TrainTime
ResNet50 (224)	78.1	4.9h	78.4	3.5h (-29%)
ResNet50 (380)	80.0	14.3h	80.3	5.8h (-59%)
ResNet152 (380)	82.4	15.5h	82.9	7.2h (-54%)
EfficientNet-B4	82.9	20.8h	83.1	9.4h (-55%)
EfficientNet-B5	83.7	42.9h	84.0	15.2h (-65%)

References

- Tan, Mingxing, et al. “EfficientNetV2: Smaller Models and Faster Training.” *International conference on machine learning*. PMLR, 2021
- Tan, Mingxing, et al. “EfficientNet: Rethinking Model Scaling for convolutional Neural Networks.” *International conference on machine learning*. PMLR, 2019

패턴인식 – 논문발표

Gradient – starvation : A Learning Proclivity in Neural Networks(NeurlPS 2021)



Gradient Starvation: A Learning Proclivity in Neural Networks

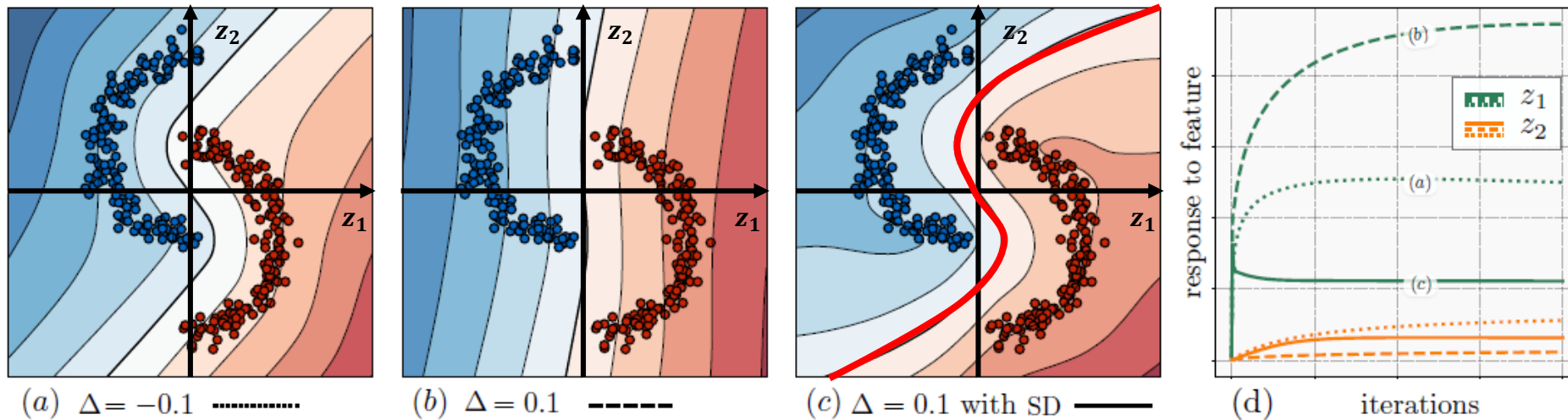
Mohammad Pezeshki^{1,2} Sékou-Oumar Kaba^{1,3} Yoshua Bengio^{1,2}
Aaron Courville^{1,2} Doina Precup^{1,3,4} Guillaume Lajoie^{1,2}

¹Mila ²Université de Montréal ³McGill University ⁴Google DeepMind
corresponding author:mohammad.pezeshki@umontreal.ca

기계공학과 2021 – 26930 유진오

Gradient starvation(GS)

- Gradient starvation : Capturing only a subset of features, when using cross entropy loss → fail to discover other predictive features
- Reason : Neural networks focus on low-level superficial correlations
→ Given strongly-correlated & fast to learn features in training data, gradient descend is biased towards them : feature imbalance
- Consequence : Lack of robustness, Excessive invariance for classification applications



Spectral Decoupling

- In Neural Tangent Kernel regime (NTK regime) output of the neural network can be approximated as 1st order linear functions of its parameters

$$\hat{\mathbf{y}}(\mathbf{X}, \boldsymbol{\theta}) = \boldsymbol{\Phi}_0 \boldsymbol{\theta}$$

Training set : $D = \{\mathbf{X}, \mathbf{y}\}, \mathbf{y} \in \{-1, +1\}^n, \boldsymbol{\theta}$: vectorized weight matrix

- Definition of GS : Feature i starves the gradient for feature j if

$$\frac{dz_j^*}{d(s_i^2)} < 0$$

$$\mathbf{Y} = \text{diag}(\mathbf{y}), \mathbf{Y}\boldsymbol{\Phi}_0 = \mathbf{U}\mathbf{S}\mathbf{V}^T, \mathbf{z} = \mathbf{U}^T \mathbf{Y} \hat{\mathbf{y}}$$

→ Where s : strength of the feature \mathbf{z} : neural network's response to a feature

- Spectral decoupling : Introduce simple regularizer (SD) to decoupling the features

→ Use SD ($= \|\hat{\mathbf{y}}\|^2$) instead of general L2 penalty ($= \|\boldsymbol{\theta}\|^2$) (α : variational parameter defined for each training example)

→ Leaving α_i independent of other $\alpha_j \neq i$ by cancel out matrix \mathbf{U}, \mathbf{S} which makes coupling terms

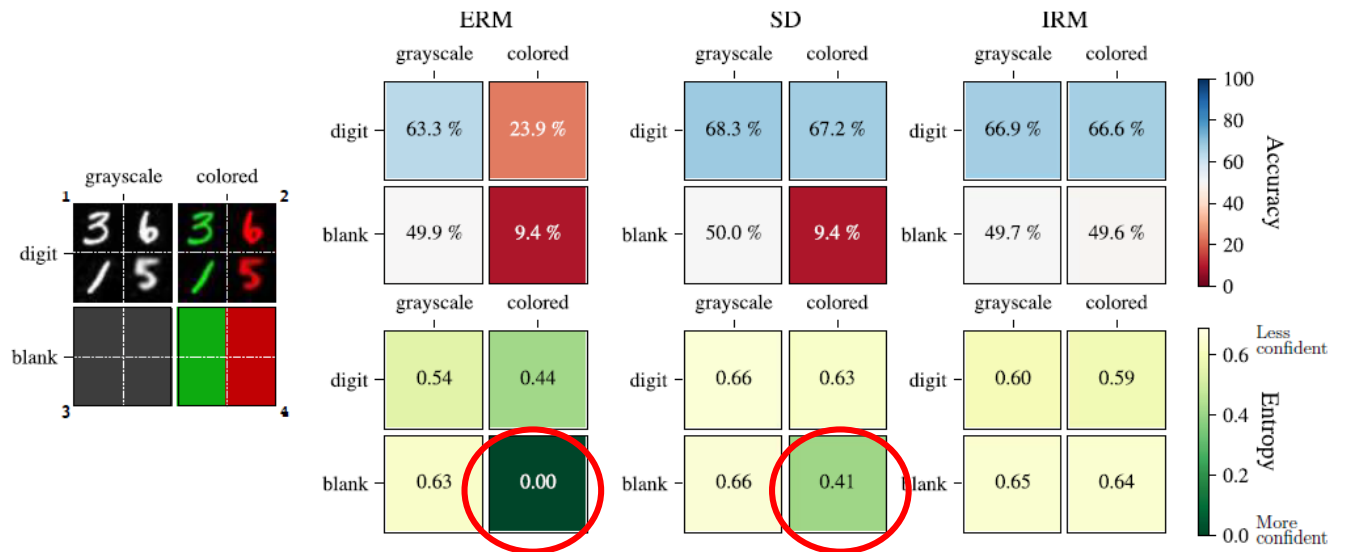
$$\text{Original : } \min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \mathbf{1} \cdot \log[1 + \exp(-\mathbf{Y}\hat{\mathbf{y}})] + \frac{\lambda}{2} \|\boldsymbol{\theta}\|^2 \rightarrow \max_{\alpha} \left(\mathbf{1} \cdot H(\alpha) - \frac{1}{2\lambda} \alpha \mathbf{Y} \boldsymbol{\Phi}_0 \boldsymbol{\Phi}_0^T \mathbf{Y}^T \alpha^T \right) \rightarrow \hat{\alpha} = \eta \left(-\log \alpha + \log(1 - \alpha) - \frac{1}{\lambda} \alpha \mathbf{U} \mathbf{S}^2 \mathbf{U}^T \right)$$

$$\text{SD : } \min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \mathbf{1} \cdot \log[1 + \exp(-\mathbf{Y}\hat{\mathbf{y}})] + \frac{\lambda}{2} \|\hat{\mathbf{y}}\|^2 \rightarrow \hat{\alpha} = \eta \left(-\log \alpha + \log(1 - \alpha) - \frac{1}{\lambda} \alpha \mathbf{U} \mathbf{S}^2 \mathbf{S}^{-2} \mathbf{U}^T \right) = \eta \left(\log \frac{1 - \alpha}{\alpha} - \frac{1}{\lambda} \alpha \right)$$

Mnist with color bias

- Task : predict binary labels $y = -1$ for digits 0 to 4 and $y = +1$ for digits 5 to 9
- A color channel(red, green) is artificially added to each example to deliberately impose a negative correlation between the color & the label
→ Color : Superficial, not robust feature
- Empirical Risk Minimization(ERM) vs SD : ERM was biased by color feature but SD wasn't → SD is more accurate and robust
(Rex and IRM requires additional multiple training environment)

Method	Train Accuracy	Test Accuracy
ERM (Vanilla Cross-Entropy)	91.1 % (± 0.4)	23.7 % (± 0.8)
REx (Krueger et al., 2020)	71.5 % (± 1.0)	68.7 % (± 0.9)
IRM (Arjovsky et al., 2019)	70.5 % (± 0.6)	67.1 % (± 1.4)
SD (this work)	70.0 % (± 0.9)	68.4 % (± 1.2)
Oracle - (grayscale images)	73.5 % (± 0.2)	73.0 % (± 0.4)
Random Guess	50 %	50 %





SHRM Laboratory for System Health and Risk Management
Department of Mechanical Engineering and Aerospace Engineering

Room 215, Building 301, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul, 08826, Republic of Korea
Tel : +82-2-880-1664, +82-2-882-1664, +82-2-883-1664, +82-2-877-1884

shrm.snu.ac.kr

THANK YOU

Training Independent Subnetworks for Robust Prediction

2021-27158 윤영석



SNU**CML**
Communications & Machine Learning Lab

TRAINING INDEPENDENT SUBNETWORKS FOR ROBUST PREDICTION

Marton Havasi*
Department of Engineering
University of Cambridge
mh740@cam.ac.uk

Rodolphe Jenatton
Google Research
rjenatton@google.com

Stanislav Fort
Stanford University
sfort1@stanford.edu

Jeremiah Zhe Liu
Google Research &
Harvard University
jereliu@google.com

Jasper Snoek
Google Research
jsnoek@google.com

Balaji Lakshminarayanan
Google Research
balajiln@google.com

Andrew M. Dai
Google Research
adai@google.com

Dustin Tran
Google Research
trandustin@google.com

Table of Contents

- Ensemble Learning
- MIMO Configurations

- Illustration of MIMO on a Synthetic Regression Example
- Loss-Landscape Analysis
- Function Space Analysis and Separation of the Subnetworks
- The Optimal Number of Subnetworks
- The Input and Batch Repetitions

- Experimental Results

Ensemble Learning

- Uncertainty estimation and out-of-distribution robustness.
 - Many real-world applications require these properties.
 - Usually achieved by using a distribution over neural networks.
 - Classic Bayesian and ensemble learning literature.
- BatchEnsemble and extensions.
 - Achieve the properties.
 - Lead to a significant computational cost.
- A multi-input multi-output configuration.
 - The insight comes from sparsity of neural network.

MIMO Configurations

- Only requires two changes to a neural network architecture.
 - Replace the input layer.
 - Replace the output layer.

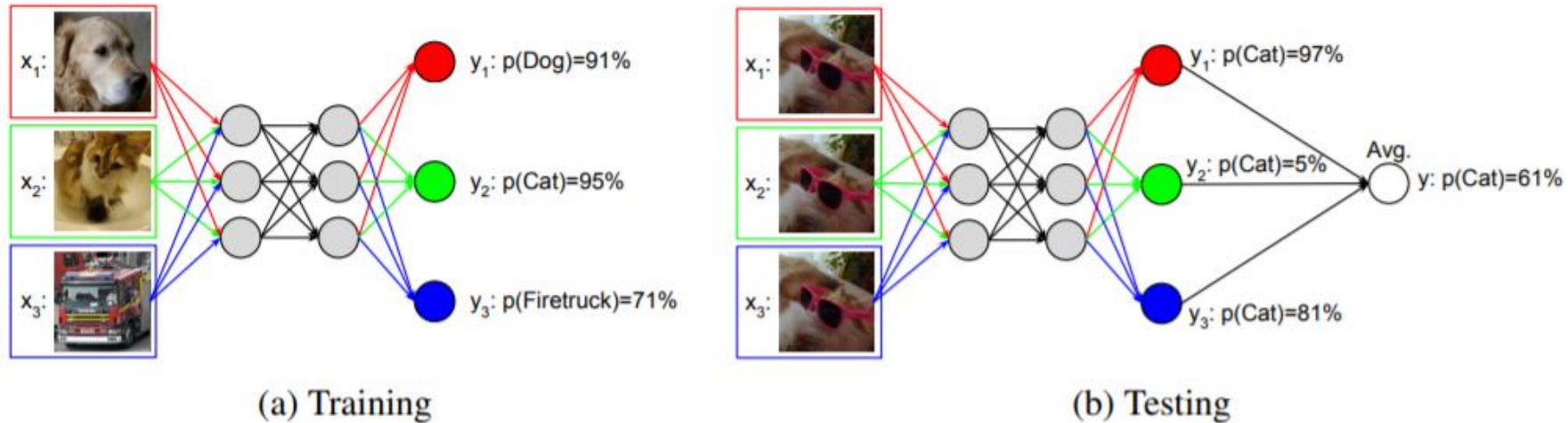


Figure 1: In the multi-input multi-output (MIMO) configuration, the network takes $M = 3$ inputs and gives M outputs. The hidden layers remain unchanged. The black connections are shared by all subnetworks, while the colored connections are for individual subnetworks. (a) During training, the inputs are independently sampled from the training set and the outputs are trained to classify their corresponding inputs. (b) During testing, the same input is repeated M times and the outputs are averaged in an ensemble to obtain the final prediction.

Illustration of MIMO on a Synthetic Regression

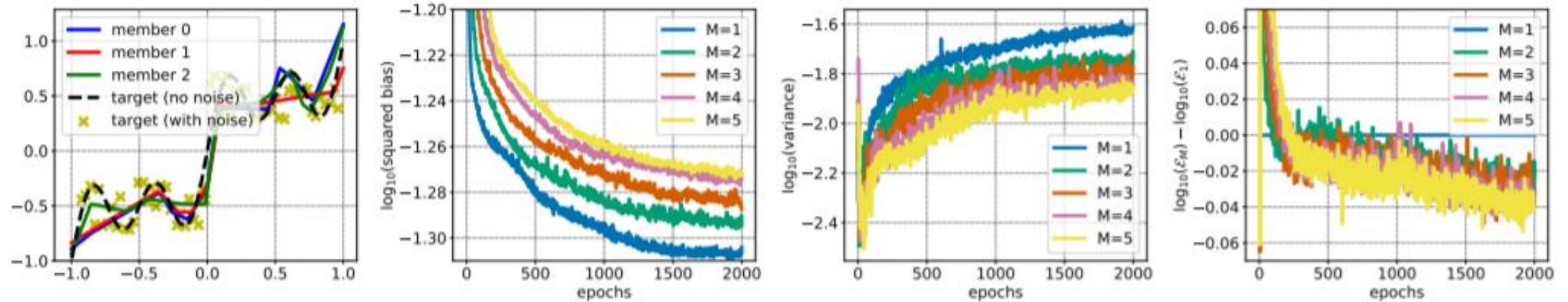


Figure 2: Illustration of MIMO applied to a synthetic regression problem. (left) Example of MIMO learning $M = 3$ diverse predictors. As M increases, predicting with MIMO comes with a higher bias but a smaller variance (two middle panels respectively). Despite the slight increase in bias, the decrease in variance translates into an improved generalization performance (right).

Loss-Landscape Analysis

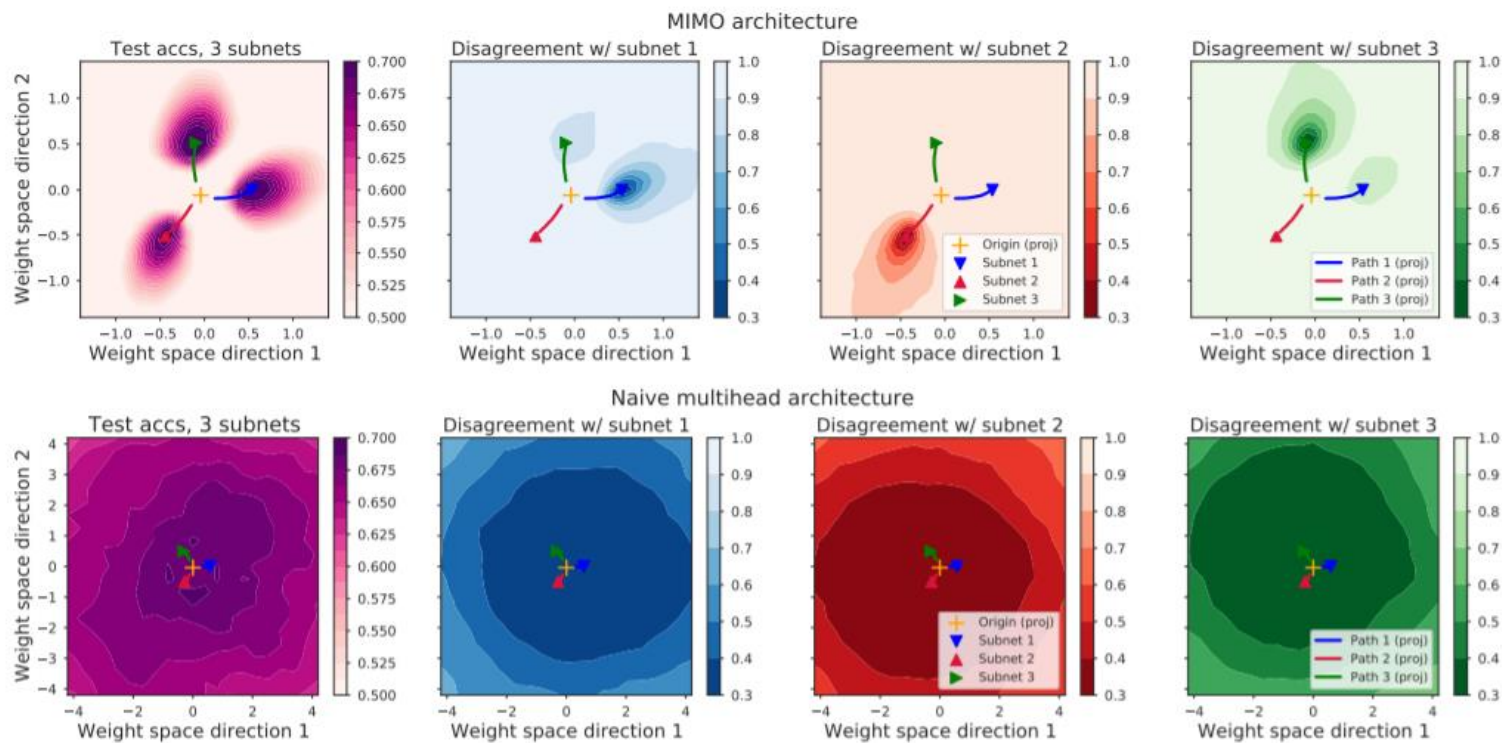


Figure 3: Accuracy landscape and function space landscape comparison of individual subnetworks for MIMO (top row) and the naive multiheaded architecture (bottom row). (left): The test accuracy in the weight space section containing $M = 3$ trained subnetworks and the origin. For the MIMO architecture, the individual subnetworks converge to three distinct low-loss basins, while naive multihead leads to the same mode. (middle-left to right): The blue, red and green panels show the disagreement between the three trained subnetworks for the same section of the weight space. For the MIMO architecture, the subnetworks often disagree, while for the naive multihead architecture they are all essentially equivalent.

Function Space Analysis and a Separation of Subnetworks

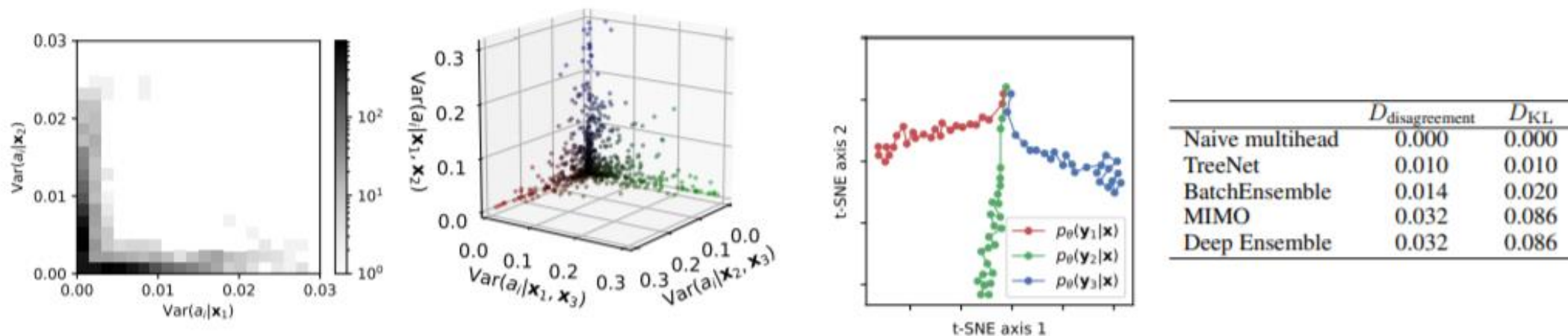


Figure 4: Analyzing the subnetworks on the CIFAR10 dataset. (left): Histogram of the conditional variances of the pre-activations w.r.t. each input ($M = 2$, ResNet28-10). (middle-left): Scatter plot of the conditional variances of the pre-activations w.r.t. each input. Almost all the pre-activations only have variance with respect to one of the inputs: the subnetwork they that are part of ($M = 3$, ResNet28-10). (middle-right): Training trajectories of the subnetworks. The subnetworks converge to different local optima ($M = 3$, SmallCNN). (right): Diversity of the members (\mathcal{D}_D) in different efficient ensemble models (ResNet 28-10).

The Optimal Number of Subnetworks

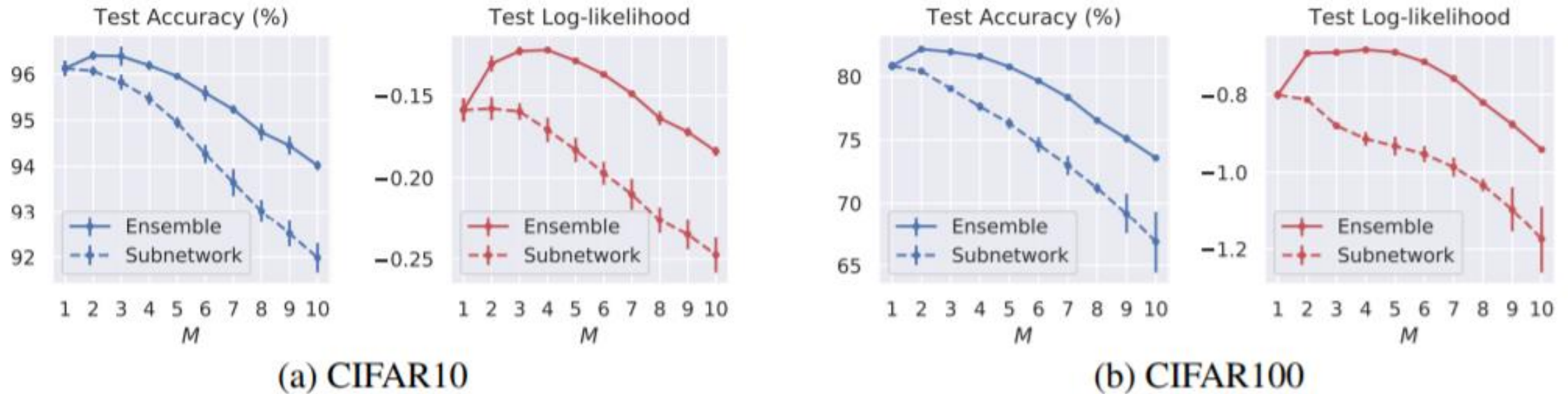
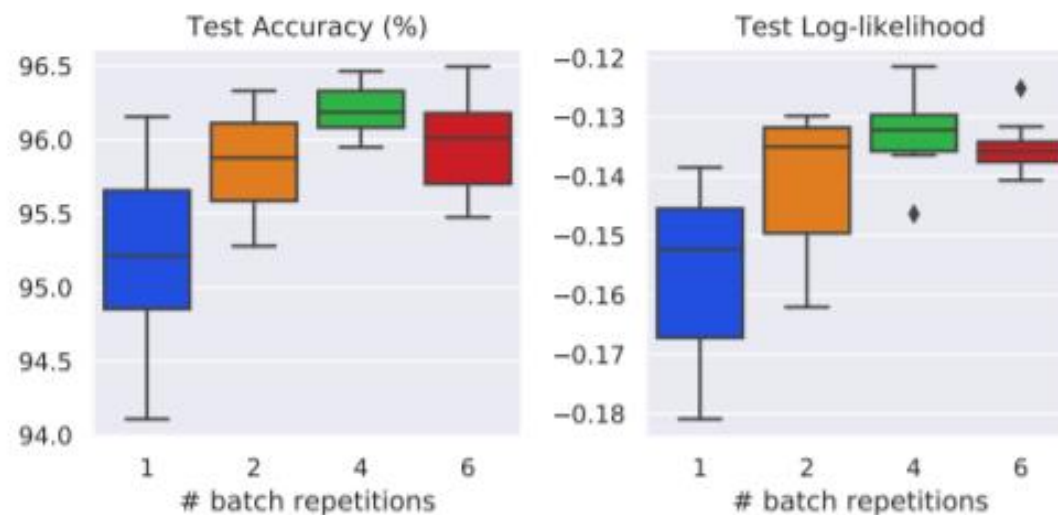


Figure 5: The performance of the subnetworks and the ensemble of the subnetworks as the number of subnetworks (M) varies. $M = 1$ is equivalent to a standard neural network (ResNet-28-10).

Input and Batch Repetitions



(a) Input repetition



(b) Batch repetition

Figure 6: (a) Performance of MIMO ($M = 2$) as a function of ρ on ImageNet. At $\rho = 0$, the subnetworks are independent and they are limited by the network capacity. With $\rho > 0$, the subnetworks are able to share features and better utilize the network capacity. Wide ResNet has $2\times$ more filters. (b) Repeating examples in the same batch improves convergence and yields a slight boost in performance.

Experimental Results

Name	Accuracy (\uparrow)	NLL (\downarrow)	ECE (\downarrow)	cAcc (\uparrow)	cNLL (\downarrow)	cECE (\downarrow)	Prediction time (\downarrow)	# Forward passes (\downarrow)
Deterministic	96	0.159	0.023	76.1	1.050	0.153	0.632	1
Dropout	95.9	0.160	0.024	68.8	1.270	0.166	0.656	1
Naive mutlihead ($M = 3$)	95.9	0.161	0.022	76.6	0.969	0.144	0.636	1
MIMO ($M = 3$) (This work)	96.4	0.123	0.010	76.6	0.927	0.112	0.639	1
TreeNet ($M = 3$)	95.9	0.158	0.018	75.5	0.969	0.137	0.961	1.5
BatchEnsemble ($M = 4$)	96.2	0.143	0.021	77.5	1.020	0.129	2.552	4
Thin Ensemble ($M = 4$)	96.3	0.115	0.008	77.2	0.840	0.089	0.823	4
Ensemble ($M = 4$)	96.6	0.114	0.010	77.9	0.810	0.087	2.536	4

Table 1: ResNet28-10/CIFAR10: The best single forward pass results are highlighted in bold.

Name	Accuracy (\uparrow)	NLL (\downarrow)	ECE (\downarrow)	cAcc (\uparrow)	cNLL (\downarrow)	cECE (\downarrow)	Prediction time (\downarrow)	# Forward passes (\downarrow)
Deterministic	79.8	0.875	0.086	51.4	2.700	0.239	0.632	1
Monte Carlo Dropout	79.6	0.830	0.050	42.6	2.900	0.202	0.656	1
Naive mutlihead ($M = 3$)	79.5	0.834	0.048	52.1	2.339	0.156	0.636	1
MIMO ($M = 3$) (This work)	82.0	0.690	0.022	53.7	2.284	0.129	0.639	1
TreeNet ($M = 3$)	80.8	0.777	0.047	53.5	2.295	0.176	0.961	1.5
BatchEnsemble ($M = 4$)	81.5	0.740	0.056	54.1	2.490	0.191	2.552	4
Thin Ensemble ($M = 4$)	81.5	0.694	0.017	53.7	2.190	0.111	0.823	4
Ensemble ($M = 4$)	82.7	0.666	0.021	54.1	2.270	0.138	2.536	4

Table 2: ResNet28-10/CIFAR100: The best single forward pass results are highlighted in bold.

Name	Accuracy (\uparrow)	NLL (\downarrow)	ECE (\downarrow)	cAcc (\uparrow)	cNLL (\downarrow)	cECE (\downarrow)	Prediction time (\downarrow)	# Forward passes (\downarrow)
Deterministic	76.100	0.943	0.039	40.500	3.200	0.105	0.640	1
Naive mutlihead ($M = 3$)	76.611	0.929	0.043	40.616	3.250	0.122	0.638	1
MIMO ($M = 2$) ($\rho = 0.6$) (This work)	77.500	0.887	0.037	43.300	3.030	0.106	0.635	1
TreeNet ($M = 2$)	78.139	0.852	0.017	42.420	3.052	0.073	0.848	1.5
BatchEnsemble ($M = 4$)	76.700	0.944	0.049	41.800	3.180	0.110	2.592	4
Ensemble ($M = 4$)	77.500	0.877	0.031	42.100	2.990	0.051	2.624	4
Wide Deterministic	77.885	0.938	0.072	45.000	3.100	0.150	1.674	1
Wide MIMO ($M = 2$) ($\rho = 0.6$) (This work)	79.300	0.843	0.061	45.791	3.048	0.147	1.706	1

Table 3: ResNet50/ImageNet: The best single forward pass results are highlighted in bold.

SITUATEDQA: Incorporating Extra-Linguistic Contexts into QA

Michael J.Q. Zhang and **Eunsol Choi**
Computer Science Department
The University of Texas at Austin
{mjqzhang, eunsol}@utexas.edu

이동렬

2021-26670

SituatedQA

- Answers to the same question may change depending on the extra-linguistic contexts (**when** and **where** the question was asked)

→ Introduce **SituatedQA**, an **open-retrieval QA dataset** where systems must produce the correct answer to a question given

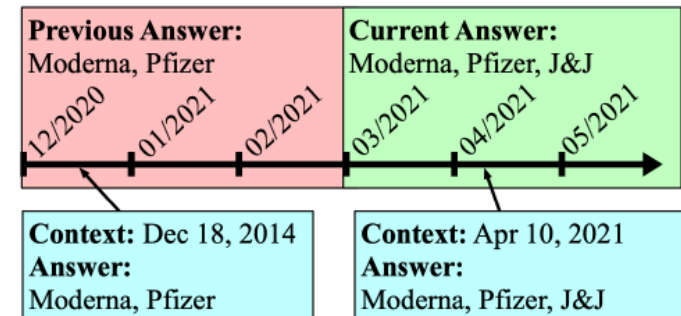
temporal (when)

or

Geographical (where) context

Context Type: Temporal

Question: Which COVID-19 vaccines have been authorized for adults in the US?



Context Type: Geographical

Question: Which COVID-19 vaccine was the first to be authorized by our government?

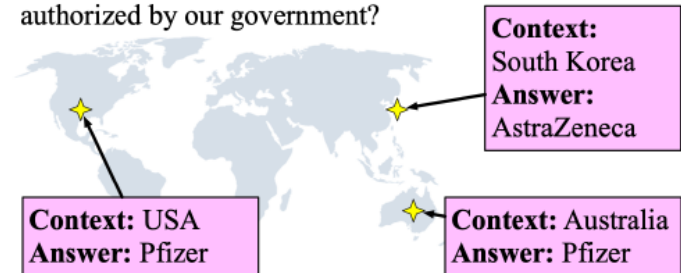


Figure 1: Examples of questions with answers that change depending on the temporal or geographical context.

SituatedQA

- For a given question q , a_i is answer when asked in the context c_i (consists of a type c_{v_t} , and a value c_{v_i})
- For temporal: each context value is timestamp (date or year)
- For geographical: each context value is geopolitical entity

Question q	Context Type c_t	Context Value c_v	Answer a
Who composed the music for the first Harry Potter film?	-	-	-
What's the biggest country in Europe excluding Russia?	-	-	-
How many seasons are there for American Horror Story?	TEMP	Sep 18, 2019 Sep 13, 2017	10 9
Who made the most three point shots in the NBA?	TEMP	2014 2005	Ray Allen Reggie Miller
When was the last time states were created?	GEO	Nigeria United States	1 October 1996 1959
Where do we rank among the world's largest cities?	GEO	Tokyo Shanghai	1st 3rd

Table 1: Examples of how questions interact with geographical and temporal context in SITUATEDQA. The first two questions are not identified as geographically nor temporally dependent.

SituatedQA : Data collection

- Data collection = identifying context-dependent questions + collecting answers from alternate contexts
1. Identification
 2. {Context / Answer} collection
 3. Validation

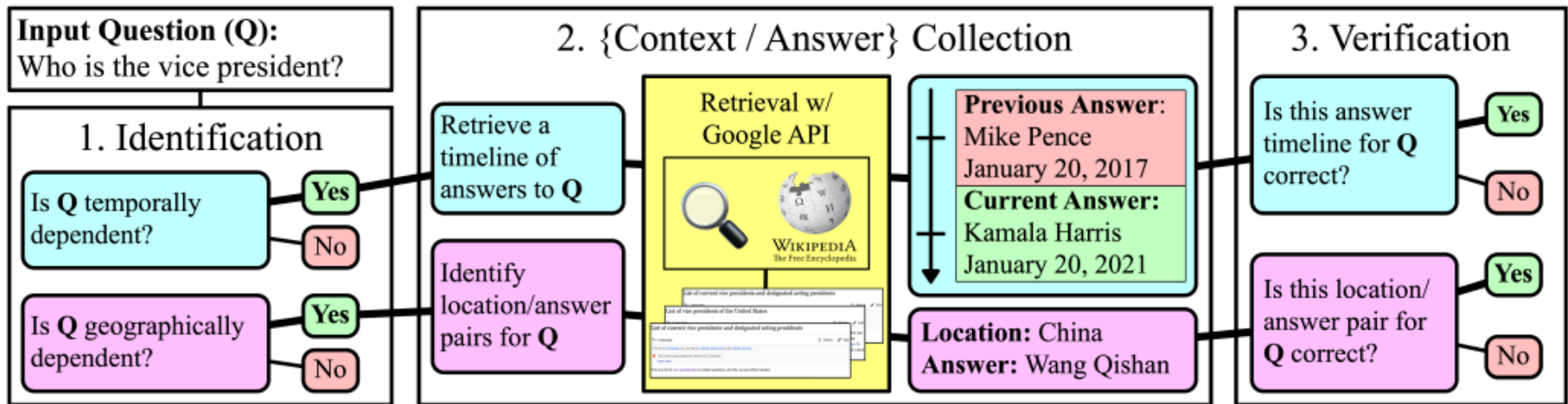


Figure 2: Data collection pipeline: Crowdworkers are first asked to identify context dependent questions. We then collect brief answer timelines for temporally dependent questions and location/answer pairs geographically dependent questions, each of which is then verified by another worker.

SituatedQA : Temporal data example

- {"question": "when did england last get to the semi final of a world cup", "id": -7924170207595297549, "edited_question": "when did england last get to the semi final of a world cup as of 2021", "date": "2021", "date_type": "sampled_year", "answer": ["2018", "2018"], "any_answer": ["1990", "2018"]}
- {"question": "when did england last get to the semi final of a world cup", "id": -7924170207595297549, "edited_question": "when did england last get to the semi final of a world cup as of 2019", "date": "2019", "date_type": "sampled_year", "answer": ["2018", "2018"], "any_answer": ["1990", "2018"]}
- {"question": "when did england last get to the semi final of a world cup", "id": -7924170207595297549, "edited_question": "when did england last get to the semi final of a world cup as of 2015", "date": "2015", "date_type": "sampled_year", "answer": ["1990", "1990"], "any_answer": ["1990", "2018"]}
- {"question": "when did england last get to the semi final of a world cup", "id": -7924170207595297549, "edited_question": "when did england last get to the semi final of a world cup as of 1998", "date": "1998", "date_type": "sampled_year", "answer": ["1990", "1990"], "any_answer": ["1990", "2018"]}
- {"question": "when did england last get to the semi final of a world cup", "id": -7924170207595297549, "edited_question": "when did england last get to the semi final of a world cup as of 2018", "date": "2018", "date_type": "start", "answer": ["2018", "2018"], "any_answer": ["1990", "2018"]}
- {"question": "when did england last get to the semi final of a world cup", "id": -7924170207595297549, "edited_question": "when did england last get to the semi final of a world cup as of 1990", "date": "1990", "date_type": "start", "answer": ["1990", "1990"], "any_answer": ["1990", "2018"]}

SituatedQA : Geographical data example

- {"question": "what is the legal amount of weed", "id": 9188546442463987107, "edited_question": "what is the legal amount of weed in maryland", "location": "maryland", "answer": ["10 grams or less"], "any_answer": ["less than 14 g (0.49 oz)", "2 ounces", "0 - illegal for recreational use", "10 grams or less", "7 grams of less"]}
- {"question": "what is the legal amount of weed", "id": 9188546442463987107, "edited_question": "what is the legal amount of weed in texas", "location": "Texas", "answer": ["0 - illegal for recreational use"], "any_answer": ["less than 14 g (0.49 oz)", "2 ounces", "0 - illegal for recreational use", "10 grams or less", "7 grams of less"]}
- {"question": "what is the legal amount of weed", "id": 9188546442463987107, "edited_question": "what is the legal amount of weed in connecticut", "location": "Connecticut", "answer": ["less than 14 g (0.49 oz)"], "any_answer": ["less than 14 g (0.49 oz)", "2 ounces", "0 - illegal for recreational use", "10 grams or less", "7 grams of less"]}
- {"question": "what is the legal amount of weed", "id": 9188546442463987107, "edited_question": "what is the legal amount of weed in washington state", "location": "washington state", "answer": ["7 grams of less"], "any_answer": ["less than 14 g (0.49 oz)", "2 ounces", "0 - illegal for recreational use", "10 grams or less", "7 grams of less"]}
- {"question": "what is the legal amount of weed", "id": 9188546442463987107, "edited_question": "what is the legal amount of weed in washington d.c.", "location": "washington d.c.", "answer": ["2 ounces"], "any_answer": ["less than 14 g (0.49 oz)", "2 ounces", "0 - illegal for recreational use", "10 grams or less", "7 grams of less"]}

Experiments

	Query Mod.	Fine-tuned	TEMP				GEO		
			Static (400)	Samp. (1472)	Start (923)	Total	Comm. (265)	Rare (240)	Total
Retrieval based			44.2	16.0	14.2	19.4	9.1	2.9	6.1
	✓		28.8	15.9	18.5	18.6	27.5	22.1	25.0
	✓	✓	39.8	17.2	24.9	23.0	27.9	25.0	26.5
Closed Book			27.2	15.3	12.9	16.2	9.4	4.6	7.1
	✓		19.5	12.4	15.7	14.5	19.2	9.2	14.5
	✓	✓	26.0	16.2	18.3	18.3	21.5	11.7	16.8
Human*			-	-	-	57.0	-	-	34.0

Table 5: Results situated question answering, reporting exact match score on the test set. In addition to reporting overall EM for each context type, we also report EM for partitions of the test set. For TEMP, we partition the test set based on how the example’s context value was generated. For GEO, we split on whether the context-value is a location is common (Comm.) or uncommon (Rare), which is determined by whether the location appears at least five times in our dataset as a geographical context.

- Exact match score
 - Retrieval based : DPR
 - Closed book : BART-large
- Lag behind human-level performance

Conclusion

- Propose the first study of how **extra-linguistic contexts** affect open retrieval QA
- Reveal that current system **fail to adapt to shifts** in the **temporal or geographical context**
- Propose the tasks and **create a dataset** for training and evaluating QA systems on modeling how facts change across contexts

감사합니다.

On the Out-of-distribution Generalization of Probabilistic Image Modeling

Fall 2021

Pattern Recognition (430.707A)

LEE, DONGJUNE

Contents

- Out-of-Distribution (OOD) Detection
- Proposed Method
- Experiment

Out of Distribution (OOD) Detection

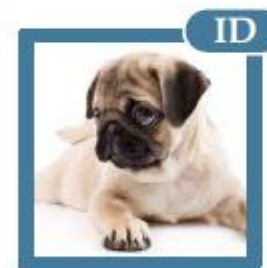
- What is OOD detection ?
 - Distinguish between in- and out-of-distribution data
- General OOD detection Methods
 - Distance-based Methods
 - **Density-based Methods**
 - Generative Models(VAE, Flow, PixelCNN...)
 - Likelihood-based score

Train



dog

Test



Out of Distribution (OOD) Detection

- A **challenge** in density-based methods
 - They often assign *higher likelihoods* to OOD data than to in-distribution data.
- **Why?**
 - *Recent Hypothesis*
 - Low-level *local features*, learned by probabilistic models, are common to images and dominate the likelihood
 - Inductive bias of the model (e.g. CNN-based).
 - Likelihood score alone is not enough for OOD detection.
- ***OOD using Product of Experts***
 - Decompose a model into two parts: Local expert & Non-local expert.
 - A new likelihood-ratio score for OOD detection.



Proposed Method

- *Main Hypothesis*
 - Low-level local features, learned by probabilistic models, are common to images and dominate the likelihood
- **Proposal**
 - Directly model the in-distribution dataset, using only local feature information.
 - A non-local(semantics) model can then be considered as the complement of a local model, from a respective full model.

Proposed Method

- Product of Experts [1]
 - Combination of multiple models of the same data by multiplying their probability distributions together and then renormalizing.
 - Two Experts: $p_l(x)$ (local expert), $p_{nl}(x)$ (non-local expert)

$$p_f(x) = \frac{p_l(x) * p_{nl}(x)}{Z}$$

- The unnormalized likelihood of a non-local model \rightarrow Proposed Score !!!

$$p_{nl}(x) \propto \frac{p_f(x)}{p_l(x)} \equiv \hat{p}_{nl}(x)$$

Full Autoregressive Model

Local Autoregressive Model

Proposed Method

- PixelCNN
- Full Autoregressive Model (full model)

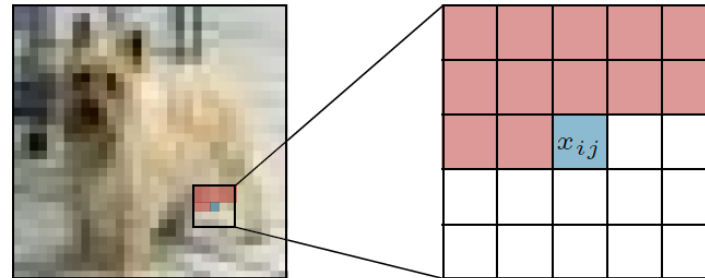
- $p_f(x) = p(x_1) \prod_{d=2}^D p(x_d | x_1, \dots, x_{d-1})$

- Local Autoregressive Model (local model)

- $p_l(x) = \prod_{i,j} p(x_{ij} | x_{[i-h:i-1, j-h:j+h]}, x_{[i, j-h:j-1]})$



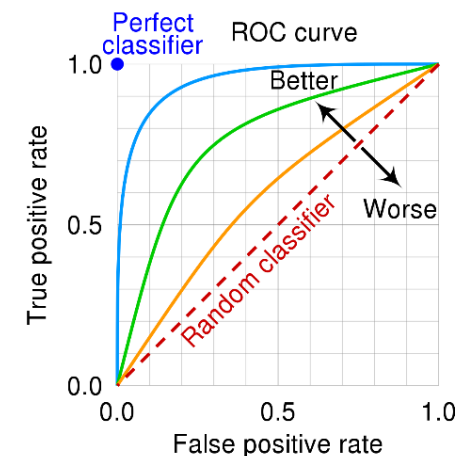
(a) Full Autoregressive Model



(b) Local Autoregressive Model

Experiment

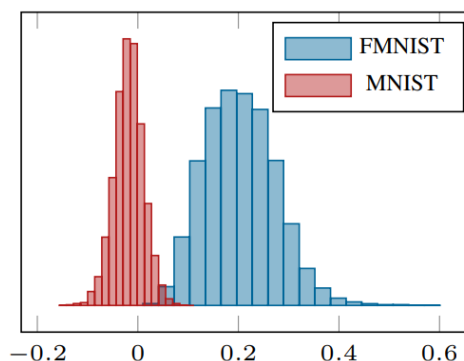
- AUROC (Area Under ROC curve)
 - A common measure for the OOD detection task
 - Formulate a curve with respect to FPR & TPR
 - The larger the better



- Result using the new score.

		Predicted condition	
		Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection

ID dataset:	FashionMNIST		CIFAR10	
OOD dataset:	MNIST	Omniglot	SVHN	CelebA
WAIC (model ensemble) [8]	0.766	0.796	1.000	-
Glow diff to PNG [43]	-	-	0.754	-
PixelCNN diff to PNG [43]	-	-	0.823	-
Likelihood Ratio in [40]	0.997	-	0.912	-
MSMA KD Tree [34]	0.693	-	0.991	-
S using Glow and FLIF [44]	0.998	1.000	0.950	0.736
S using PCNN and FLIF [44]	0.967	1.000	0.929	0.535
Full PixelCNN likelihood ^b	0.074	0.361	0.113	0.602
Our method	1.000	1.000	0.969	0.949



<i>False Positive Rate</i>	$\frac{FP}{FP + TN}$
<i>True Positive Rate</i>	$\frac{TP}{TP + FN}$

Thank You