local plasticity rules can learn deep representations using self-supervised contrastive predictions

NeurIPS 2021

Bernd Illing **Jean Ventura**

Guillaume Bellec* **Wulfram Gerstner***

Department of Computer Science & Department of Life Sciences
École Polytechnique Fédérale de Lausanne
1015 Switzerland

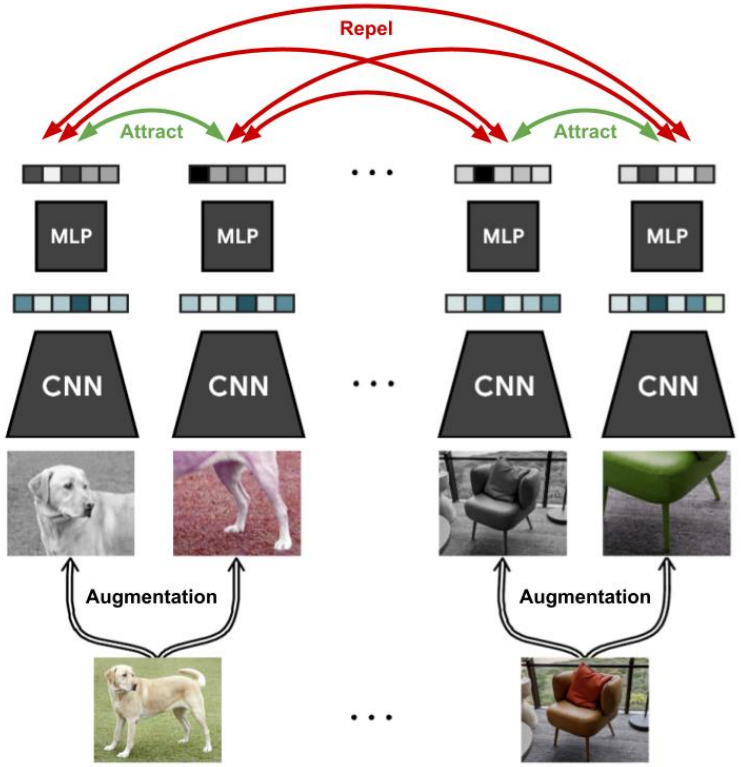
Goal : How brain learns deep hierarchical representations

- What is Biologically Plausible Learning? : Learning under biological constraints
- What is biological Constraints?
  - No label is given (unsupervised)
  - Input data is sequential form (online learning)
  - Weight updates depend on local and recent neural activity (Can't use backpropagation of gradients[1])
  - Modulated by broadcast signal related to reward and attention[2]
  - Follows local plasticity rule from neuroscience (Hebbian update rule[3])

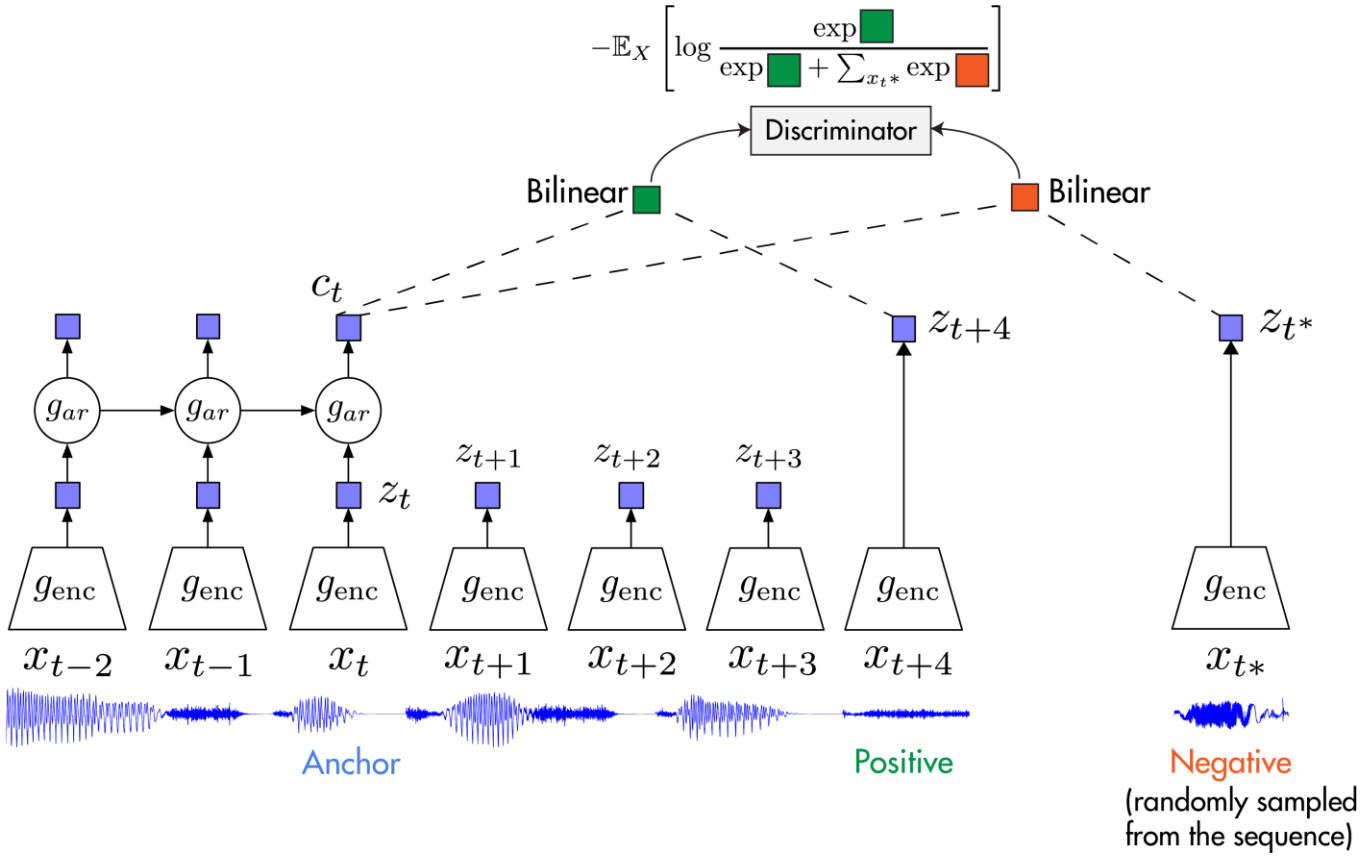
- Currently studied Biologically Plausible implementations of back-propagation : has limitations
  - Neuron-specific error signal needs to be transmitted by a separate error network[4][5](FA, TP)
  - Use only local activity but they require to wait for convergence to an equilibrium[6][7](EP, PC)

- Propose learning rule satisfies biological Constraints & build deep hierarchical representations
  - Temporal structure of natural stimuli is a rich source of information
  - Self-supervised learning (contrastive predictive coding[8] frame) from temporal data

Background



Contrastive learning

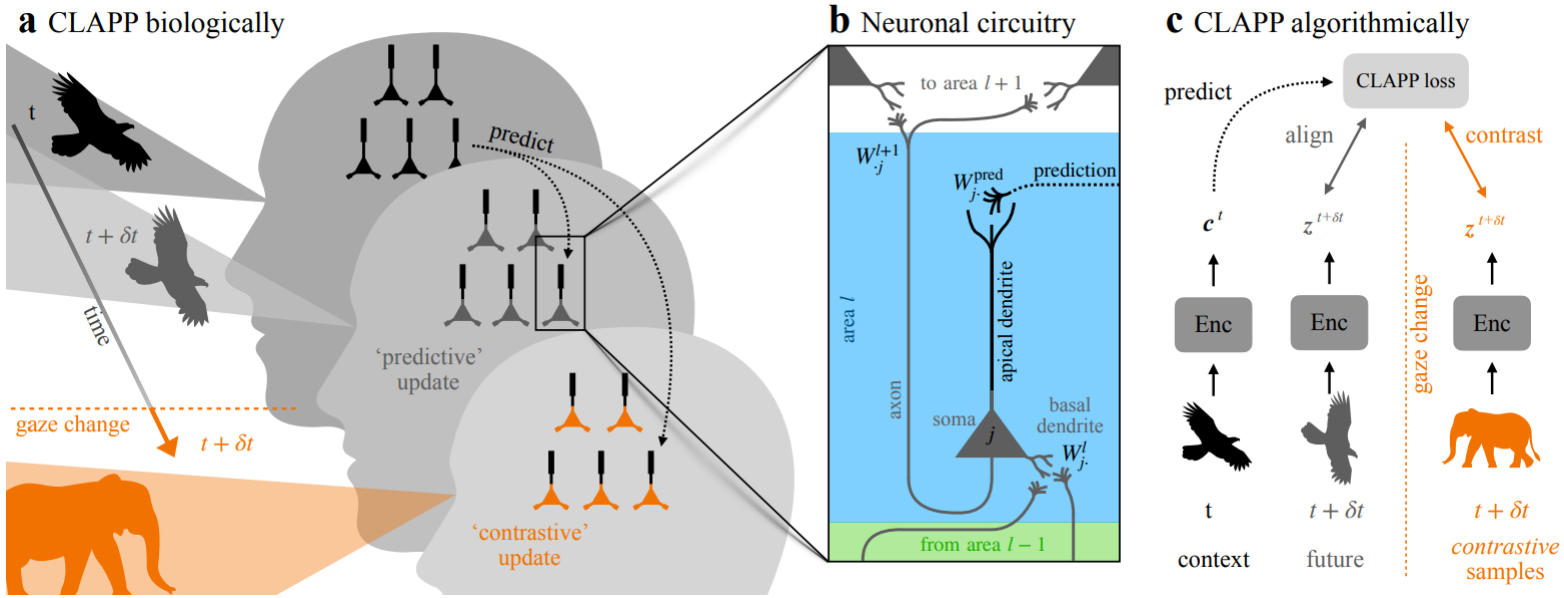


Contrastive predictive coding (2018)

→ Both need past information of large negative samples

Idea & model

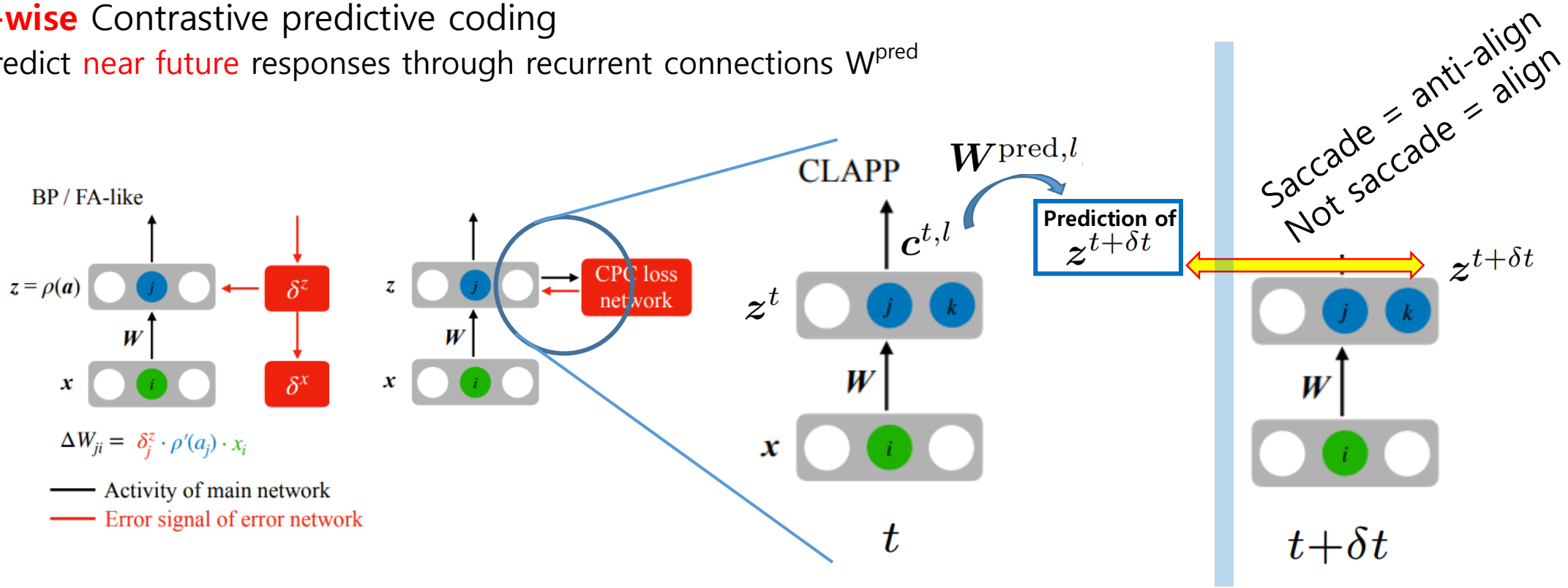
- Inspiration from deep self-supervised learning algorithms
  - How to define positive & negative sample?



- Brain is self-aware of typical, self-generated changes of gaze direction ('saccades')
  - Distinguish input from input arriving after a saccade towards a new object
  - Broadcast signal modulates plasticity (assign positive & negative sample)

Idea & model

- Weight update rule
  - **Layer-wise** Contrastive predictive coding
    - Predict near future responses through recurrent connections W^{pred}



score function : 'guess' performed fixation or saccade

$$u_t^{t+\delta t,l} = z^{t+\delta t,l\top} W^{\text{pred},l} c^{t,l}$$

$$\mathcal{L}_{CLAPP}^{t,l} = \max\left(0, 1 - y^t \cdot u_t^{t+\delta t,l}\right) \quad \text{with} \quad \begin{cases} y^t = +1 & \text{for fixation} \\ y^t = -1 & \text{for saccade} \end{cases}$$

Layer-wise Hinge loss

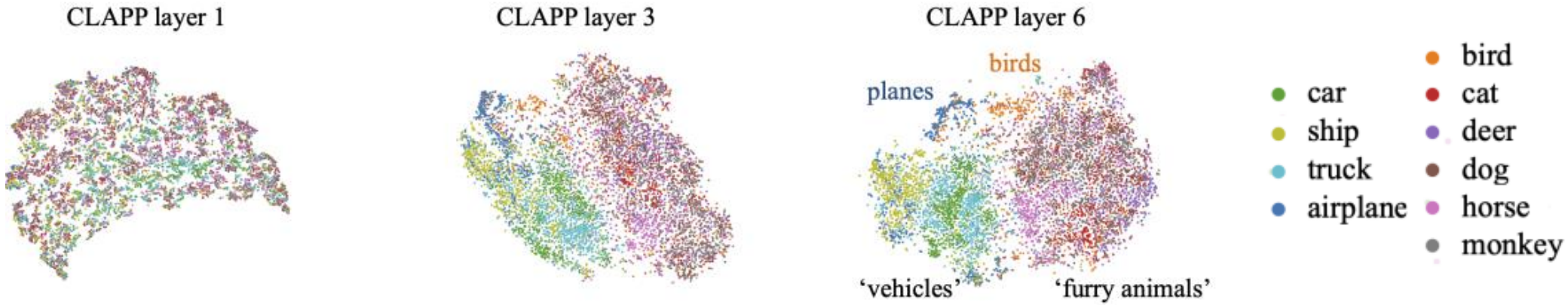
$$\Delta W_{ji}^t = \underbrace{\gamma_t}_{\text{broadcast factors}} \cdot \underbrace{(W^{\text{pred}} c^{t-\delta t})_j}_{\text{dendritic prediction}} \cdot \underbrace{\rho'(a_j^t) x_i^t}_{\text{local activity}}$$

$$\Delta W_{km}^{c,t} = \underbrace{\gamma_t}_{\text{broadcast factors}} \cdot \underbrace{(W^{\text{retro}} z^t)_k}_{\text{dendritic retrodiction}} \cdot \underbrace{\rho'(a_k^{c,t-\delta t}) x_m^{c,t-\delta t}}_{\text{local activity}}$$

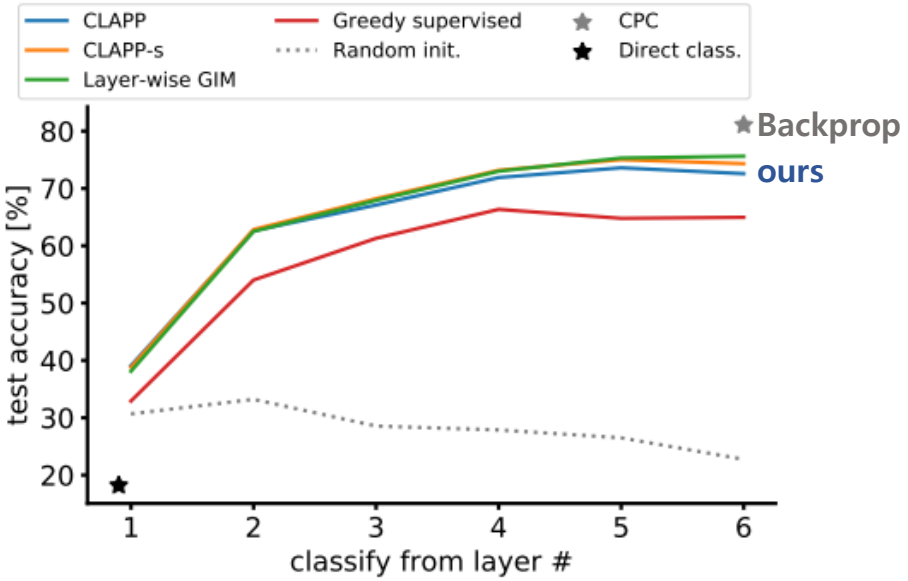
→ Local Constraints 만족

Experiments

- t-SNE projection of neuronal activities



- class labels



Method	local in ... space?	local in ... time?	STL-10	LibriSpeech	UCF-101
Chance performance			10.0	2.4	0.99
Random init.	✓	✓	21.8	27.7*	30.5
MFCC	✓	✓	-	39.7*	-
Greedy supervised	(✓)	✓	66.3	73.4*	-
Supervised	✗	✓	73.2	77.7*	51.5
CPC	✗	✗	81.1	64.3	35.7
Layer-wise GIM	✗	✗	75.6	63.9	41.2
Hinge Loss CPC (ours)	✗	✗	80.3	62.8	36.1
CLAPP-s (2 modules of 3 layers)	✗	✗	77.6	-	-
CLAPP-s (3 modules of 2 layers)	✗	✗	77.4	-	-
CLAPP-s (ours)	✓	✗	75.0	61.7	41.6
time-local Hinge Loss CPC (ours)	✗	✓	79.1	-	-
CLAPP (ours)	✓	✓	73.6	-	-

[1] T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, and G. Hinton. Backpropagation and the brain. Nature Reviews Neuroscience, 21(6):335–346, 2020.
[2] W. Gerstner, M. Lehmann, V. Liakoni, D. Corneil, and J. Brea. Eligibility traces and plasticity on behavioral time scales: experimental support of neohebbian three-factor learning rules. Frontiers in neural circuits, 12:53, 2018.
[3] H. Markram, W. Gerstner, and P. J. Sjöström. A history of spike-timing-dependent plasticity. Frontiers in synaptic neuroscience, 3:4, 2011.
[4] T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. Nature communications, 7(1):1–10, 2016.
[5] D. Kunin, A. Nayebi, J. Sagastuy-Brena, S. Ganguli, J. M. Bloom, and D. L. K. Yamins. Two Routes to Scalable Credit Assignment without Weight Symmetry. In ICML, 2020. URL http: //arxiv.org/abs/2003.01513.
[6] B. Scellier and Y. Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. Frontiers in computational neuroscience, 11:24, 2017.
[7] X. Xie and H. S. Seung. Equivalence of backpropagation and contrastive hebbian learning in a layered network. Neural computation, 15(2):441–454, 2003.
[8] Van den Oord, Y. Li, and O. Vinyals. Representation Learning with Contrastive Predictive Coding. arXiv Prepr., 2018.

Self-supervised Graph-level Representation Learning with Local and Global Structure

Minghao Xu [1] **Hang Wang** [1] **Bingbing Ni** [1] **Hongyu Guo** [2] **Jian Tang** [3 4 5]

Accepted as short talk at ICML 2021

2021 Nov. 구정현

Preliminaries

- Self-supervised Learning
- Graph Neural Network(GNN)
- Expectation-maximization(EM) algorithm

Motivation

- Learning informative representation of whole graphs is a fundamental problem in a variety of domains.

- GNNs have been successful in some domains and tasks, mainly in supervised fashion.

→ However, since many scientific domains lack labeled data, it is becoming increasingly important to learn in unsupervised or self-supervised fashion.

- There are some recent works that learn graph representation in a self-supervised manner.

→ But they fail to discover the global-semantic feature

Introduction to GraphLoG

Local instance + Global semantic structure

- Preserve local instance structure
  - Similar graphs are embedded close to each other and dissimilar ones stay far apart
- Reflect the global semantic structure with EM algorithm
  - Graphs with similar semantic properties are compactly embedded, and constitute hierarchical semantic clusters

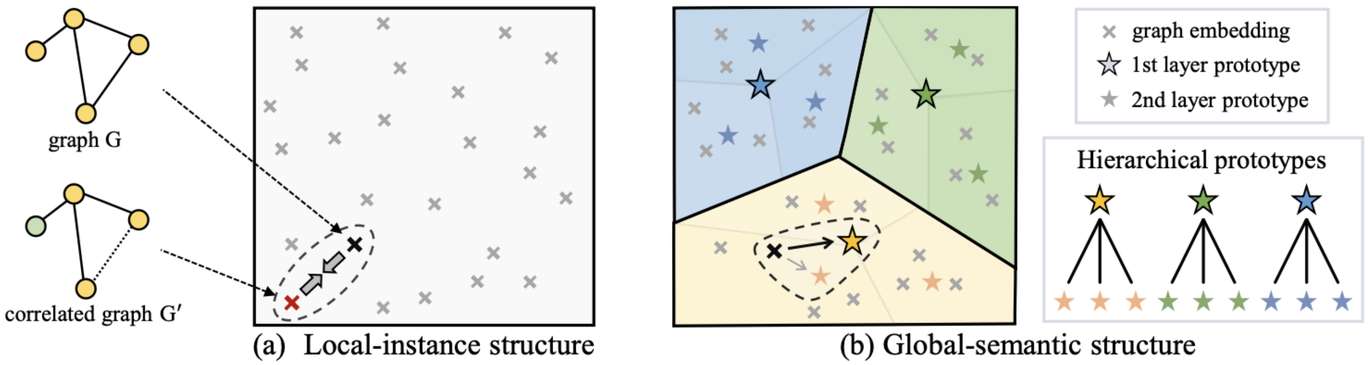
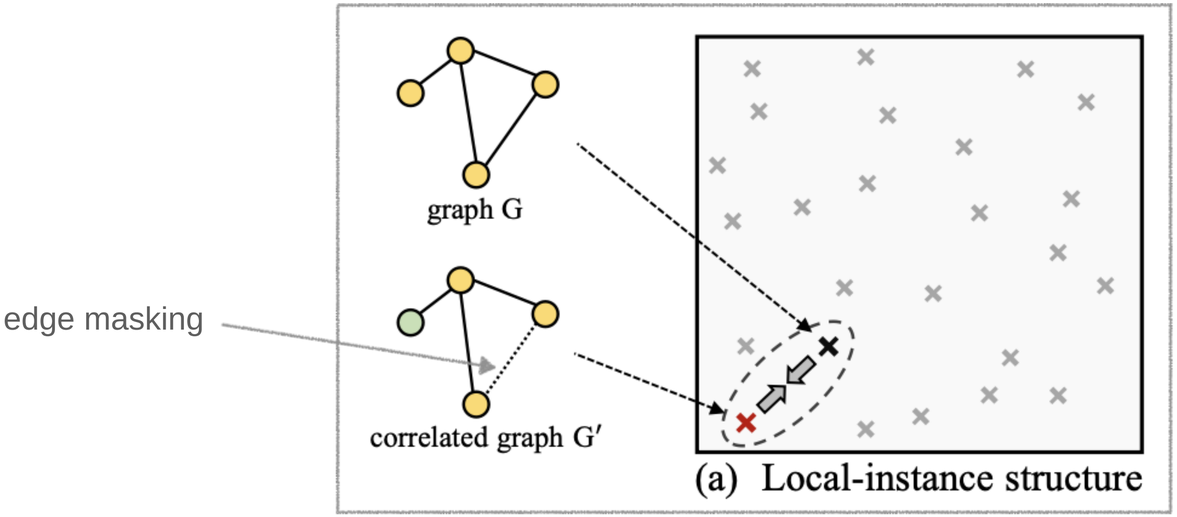


Figure 1. **Illustration of GraphLoG.** (a) Correlated graphs are constrained to be adjacently embedded to pursue the local-instance structure of the data. (b) Hierarchical prototypes are employed to discover and refine the global-semantic structure of the data.

Deep-dive

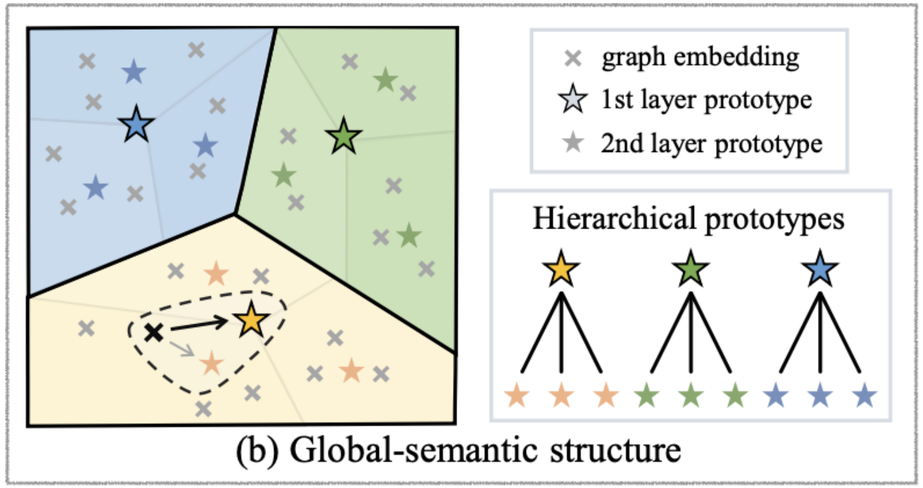
Local-instance learning



- **Finding correlated pairs(graph, subgraph)**
  - Graph pair:
    - Given a graph $G = (V, E, X_V, X_E)$, its correlated counterpart $G' = (V', E', X_{V'}, X_{E'})$ is obtained through randomly masking a part of node/edge attributes in the graph
  - Subgraph pair:
    - For a subgraph G_v constituted by node v and its L-hop neighborhoods in graph G, regard the corresponding subgraph G'_v in graph G' as its correlated counterpart.
- **Finding negative pairs**
  - Given a correlated graph/subgraph pair, substitute G(graph) or G_v(subgraph) randomly with another graph or a subgraph centered around another node to construct negative pairs
- **Objective function for local-instance learning**
  - $\min_\theta \mathcal{L}_{local} = \min_\theta (\mathcal{L}_{graph} + \mathcal{L}_{sub})$
    - $\mathcal{L}_{graph} = \mathbb{E}_{(G_+, G'_+) \sim p(G,G'),(G_-,G'_-) p_n(G,G')}[s(G_+, G'_+) - s(G_-, G'_-)]$
    - $\mathcal{L}_{sub} = \mathbb{E}_{(G_u, G'_u) \sim p(G_v, G'_v),(G_v, G'_w) p_n(G_v, G'_v)}[s(G_u, G'_u) - s(G_v, G'_w)]$
    - $s(x, y) = x^\top y / \| x \| \| y \|$: cosine similarity

Deep-dive

Global semantic learning by EM algorithm



- **Basic idea**
  - Since the latent variables are not given, it is hard to directly maximize the likelihood function $p(G, Z|\theta, C)$ → utilize EM algorithm
- **Initialization of model parameters**
  - After pre-training the GNN by minimizing $\mathcal{L}_{local}$, K-means clustering is applied layer-by-layer.
- **E-step**
  - Construct the lower-bound of the log-likelihood
  - Sample latent variables with the parameters of previous iteration
  - The posterior distribution of Z: $p(Z|G, \theta_{t-1}, C_{t-1})$
- **M-step**
  - Maximize the expected log-likelihood with respect to the posterior distribution of latent variables
  - $Q(\theta, C) = \mathbb{E}_{p(Z|G,\theta_{t-1},C_{t-1})}[\log p(G, Z|\theta, C)]$
  - $\theta^t = \underset{\theta}{\operatorname{argmax}} Q(\theta, C)$, $C^t = \underset{C}{\operatorname{argmax}} Q(\theta, C)$

Algorithm

Algorithm 1 Optimization Algorithm of GraphLoG.

Input: Unlabeled graph data set $\mathbf{G}$, the number of learning steps T.
Output: Pre-trained GNN model GNN_{θ_T}.
Pre-train GNN with local objective function (Eq. 9).
Initialize model parameters θ_0 and $\mathbf{C}_0$.
for $t = 1$ **to** T **do**
  Sample a mini-batch $\widetilde{\mathbf{G}}$ from $\mathbf{G}$.
  $\diamondsuit$ *E-step*:
  Sample latent variables $\widetilde{\mathbf{Z}}_{est}$ with $\text{GNN}_{\theta_{t-1}}$ and $\mathbf{C}_{t-1}$.
  $\diamondsuit$ *M-step*:
  Update model parameters:

$$\theta_t \leftarrow \theta_{t-1} - \nabla_\theta(\mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}}),$$
$$\mathbf{C}_t \leftarrow \mathbf{C}_{t-1} - \nabla_{\mathbf{C}}(\mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}}).$$

end for

Results

Graph Embeddings

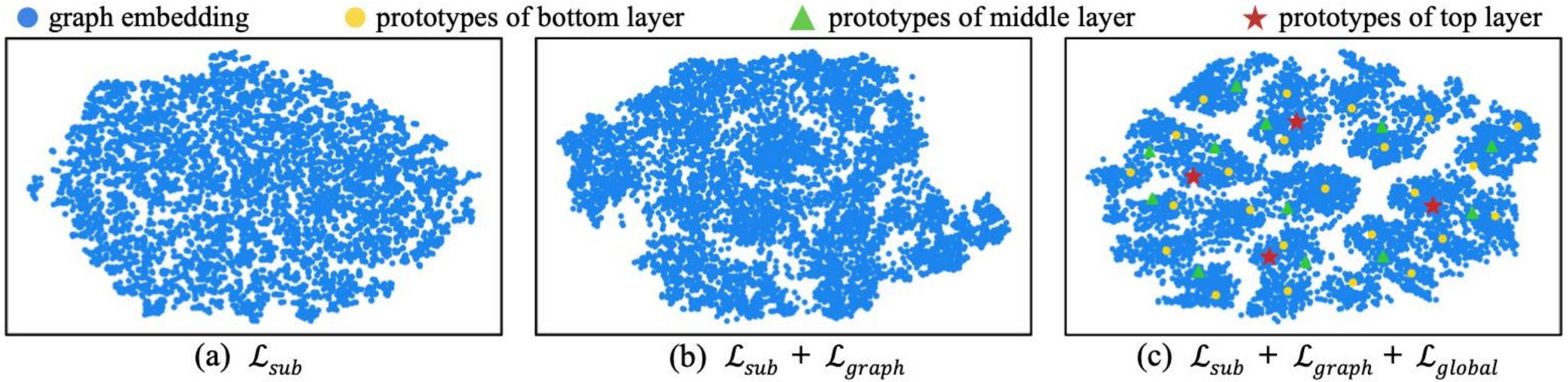


Figure 2. The t-SNE visualization on ZINC15 database (*i.e.* the pre-training data set for chemistry domain).

Results

Chemistry and Biology domain

Table 1. Test ROC-AUC (%) on downstream molecular property prediction benchmarks.

Methods	BBBP	Tox21	ToxCast	SIDER	ClinTox	MUV	HIV	BACE	Avg
Random	65.8 ± 4.5	74.0 ± 0.8	63.4 ± 0.6	57.3 ± 1.6	58.0 ± 4.4	71.8 ± 2.5	75.3 ± 1.9	70.1 ± 5.4	67.0
EdgePred (2016)	67.3 ± 2.4	76.0 ± 0.6	64.1 ± 0.6	60.4 ± 0.7	64.1 ± 3.7	74.1 ± 2.1	76.3 ± 1.0	79.9 ± 0.9	70.3
InfoGraph (2019)	68.2 ± 0.7	75.5 ± 0.6	63.1 ± 0.3	59.4 ± 1.0	70.5 ± 1.8	75.6 ± 1.2	77.6 ± 0.4	78.9 ± 1.1	71.1
AttrMasking (2019)	64.3 ± 2.8	**76.7** ± 0.4	**64.2** ± 0.5	61.0 ± 0.7	71.8 ± 4.1	74.7 ± 1.4	77.2 ± 1.1	79.3 ± 1.6	71.1
ContextPred (2019)	68.0 ± 2.0	75.7 ± 0.7	63.9 ± 0.6	60.9 ± 0.6	65.9 ± 3.8	75.8 ± 1.7	77.3 ± 1.0	79.6 ± 1.2	70.9
GraphPartition (2020b)	70.3 ± 0.7	75.2 ± 0.4	63.2 ± 0.3	61.0 ± 0.8	64.2 ± 0.5	75.4 ± 1.7	77.1 ± 0.7	79.6 ± 1.8	70.8
GraphCL (2020a)	69.5 ± 0.5	75.4 ± 0.9	63.8 ± 0.4	60.8 ± 0.7	70.1 ± 1.9	74.5 ± 1.3	77.6 ± 0.9	78.2 ± 1.2	71.3
GraphLoG (ours)	**72.5** ± 0.8	75.7 ± 0.5	63.5 ± 0.7	**61.2** ± 1.1	**76.7** ± 3.3	**76.0** ± 1.1	**77.8** ± 0.8	**83.5** ± 1.2	**73.4**

- Pre-training: self-supervised learning

  DB: ZINC15 with 2M unlabeled molecules

- Down-stream task: 8 binary classification

  DB: MoleculeNet

Table 2. Test ROC-AUC (%) on downstream biological function prediction benchmark.

Methods	ROC-AUC (%)
Random	64.8 ± 1.0
EdgePred (Kipf & Welling, 2016)	70.5 ± 0.7
InfoGraph (Sun et al., 2019)	70.7 ± 0.5
AttrMasking (Hu et al., 2019)	70.5 ± 0.5
ContextPred (Hu et al., 2019)	69.9 ± 0.3
GraphPartition (You et al., 2020b)	71.0 ± 0.2
GraphCL (You et al., 2020a)	71.2 ± 0.6
GraphLoG (ours)	**72.9** ± 0.7

- Pre-training: 395K unlabeled protein ego-network
- Downstream task: prediction of 40 fine-grained biological functions of 8 species

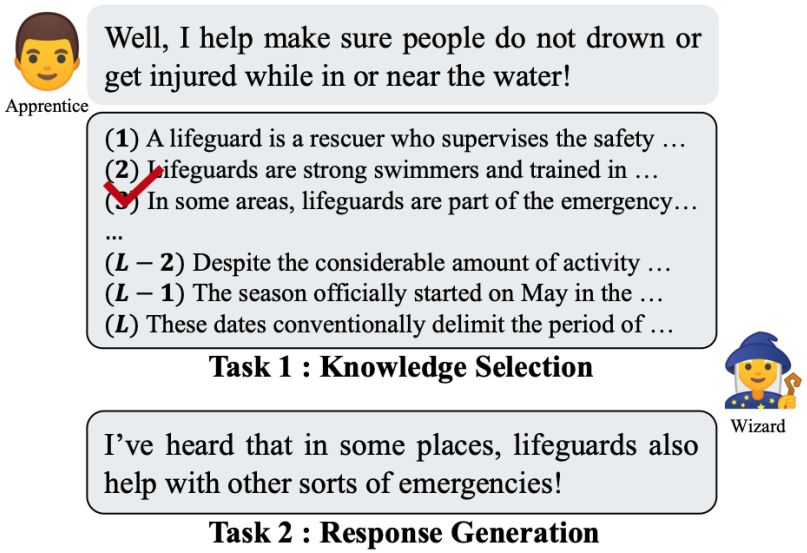
2021 EMNLP

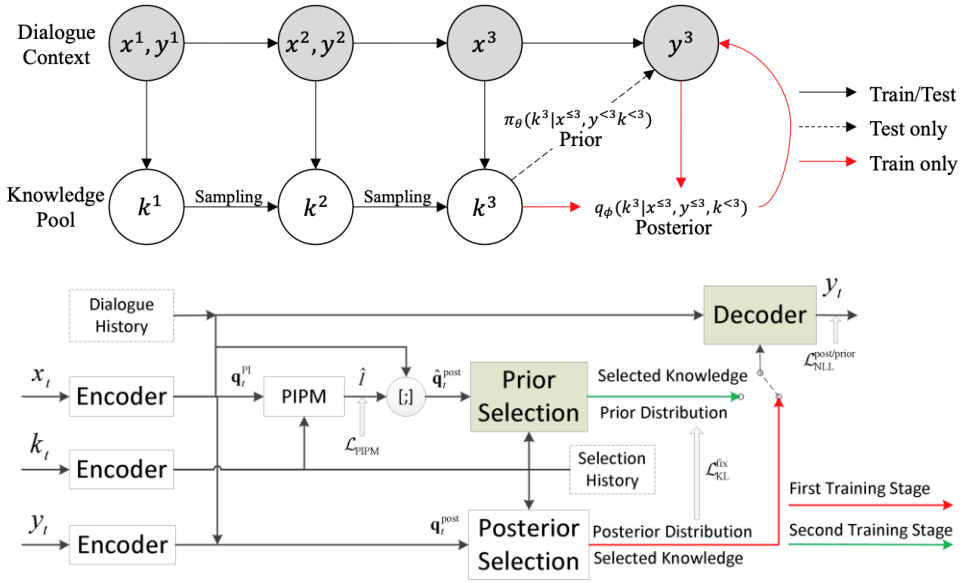
CoLV: A Collaborative Latent Variable Model for Knowledge-Grounded Dialogue Generation

2021-20669 인공지능 전공

MINBEOM KIM

Knowledge-Grounded Dialogue





Based on dialogue context,

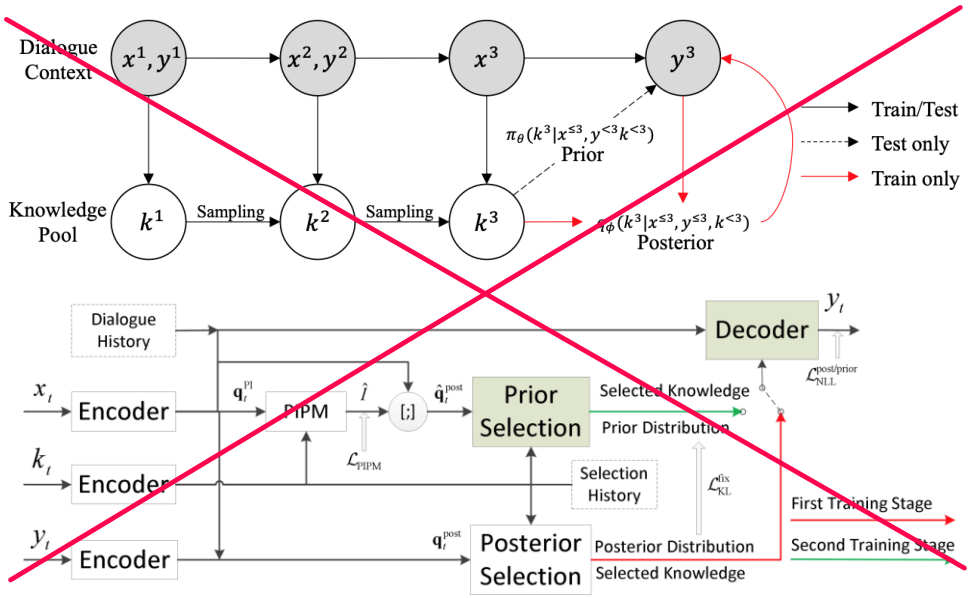
Task 1 : Knowledge Selection => Task 2 : Response Generation, 2-phase task.

figure 1, 2. Sequential Latent Knowledge Selection for Knowledge Grounded Dialogue, Kim et al.

figure 3. Bridging the Gap between Prior and Posterior Knowledge Selection for Knowledge-Grounded Dialogue Generation, Chen et al.

Knowledge-Grounded Dialogue

Dialogue context	What is your favorite number? → I love the number 7. What do you think about that?
Knowledge candidates	**1**. Anyone who dares to kill Cain "will suffer vengeance seven times over". **2**. Seven is the natural number following six and preceding eight. **3**. Islam first came to the western coast when Arab traders as early as the 7th century CE. **4**. The number 7 has been associated with a great deal of symbolism in religion. In western culture, it is often considered lucky. ..... **N**. This genre has been popular throughout the history of culture.
Response a	Yeah. I know that it is before 8 and after 6!
Response b	Yes, it is known as a lucky number in western countries!
Response c	I think 7 is lucky certain cultures. It also depicts some religious importance.



Deny 2-phase learning methodology!

Motivation : Knowledge and Response should be considered jointly

figure 4. CoLV: A Collaborative Latent Variable Model for Knowledge-Grounded Dialogue Generation

Motivation

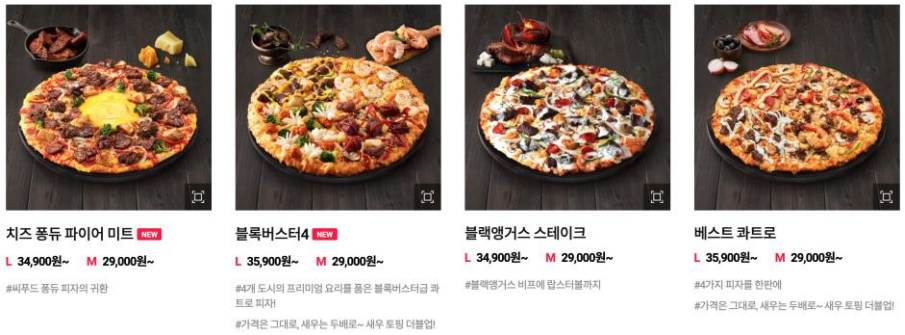
Question : 우리 오늘 저녁 머먹지??

식당	아침	점심	저녁
소담마루(880-8698)		당분간 폐점	당분간 폐점
라운지오(882-7005)		피자, 스파게티류 ※운영시간: 11:00~19:00 ※혼잡시간: 12:00~12:30	피자, 스파게티류 ※운영시간: 11:00~19:00
기숙사식당(881-9072)	조갯살미역국백반&꽁치조림(#) 3,500원 ※운영시간: 08:00-09:30	어묵매운탕(#) 3,000원 규동 4,000원 ※운영시간: 11:30~13:30	오므라이스&소시지구이 4,000원 ※운영시간: 17:30~19:00

Knowledge 1 : SNU menu

Response 1 : 피자땡기는데 피자 고??

Response 2 : 그냥 가까운 기숙사식당 가자



Knowledge 2 : Domino menu

Response 1 : 피자땡기는데 피자 고??

Response 2 : 기름진거 말고 다른 메뉴 보자

Knowledge and Response should be considered jointly !

Methodology

Contribution Point 1!

1. Task 1,2 output : k, r is discrete.

2. For jointly learning, represent k, r as $\mathbf{z_k}$, $\mathbf{z_r}$

=> Represent as continuous vector!

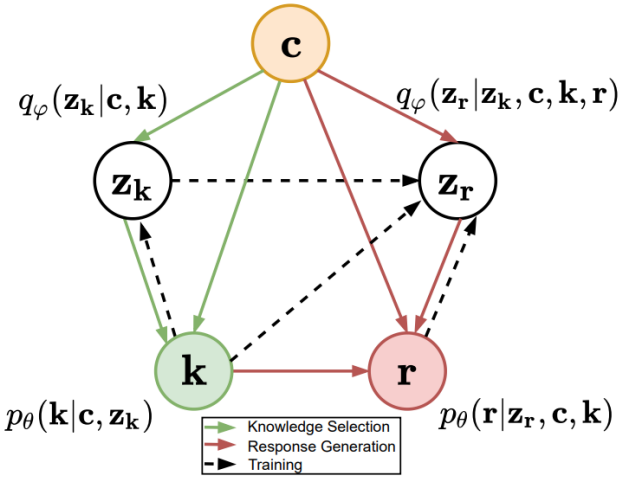


Figure 1: The graphical framework for CoLV model. $\mathbf{c}$: dialogue context, $\mathbf{k}$: knowledge, $\mathbf{r}$: response. The dotted line denotes training procedure solely, while the solid line denotes both training and inference process.

$$\mathcal{L}_{\text{CoLV}} =$$
$$- KL(q_\varphi(\mathbf{z_k}|\mathbf{c},\mathbf{k})||p_\phi(\mathbf{z_k}|\mathbf{c}))$$
$$- KL(q_\varphi(\mathbf{z_r}|\mathbf{z_k},\mathbf{c},\mathbf{k},\mathbf{r})||p_\phi(\mathbf{z_r}|\mathbf{z_k},\mathbf{c}))$$
$$+ \mathbb{E}_{\mathbf{z_k}\sim q_\varphi}[\log p_\theta(\mathbf{k}|\mathbf{z_k},\mathbf{c})]$$
$$+ \mathbb{E}_{\mathbf{z_r}\sim q_\varphi}[\log p_\theta(\mathbf{r}|\mathbf{z_r},\mathbf{k},\mathbf{c})],$$

$$p_\theta(\mathbf{k},\mathbf{r}|\mathbf{c}) = \int_{\mathbf{z_k}} \sum_{\mathbf{z_r}} p_\theta(\mathbf{k}|\mathbf{z_k},\mathbf{c}) p_\theta(\mathbf{r}|\mathbf{z_r},\mathbf{k},\mathbf{c}) \cdot p_\phi(\mathbf{z_r}|\mathbf{z_k},\mathbf{c}) p_\phi(\mathbf{z_k}|\mathbf{c}) d\mathbf{z_k},$$

Alingment k,r to $\mathbf{z_k}$, $\mathbf{z_r}$

Existing sota Supervised Learning!

such as SKT(kim et al., 2020), PIPM(Chen et al., 2020)

Methodology

Contribution Point 2!

1. $\mathbf{z_k}$, $\mathbf{z_r}$ should take diverse relation between k and z
2. Not 2-phase, end-to-end learn with continuous vector
3. Give Gaussian prior for $\mathbf{z_k}$ to diverse inference

$$p_\phi(\mathbf{z_k}|\mathbf{c}) = \mathrm{Cat}_\phi(\mathbf{z_k}|\pi),$$
$$p_\phi(\mathbf{z_r}|\mathbf{z_k}, \mathbf{c}) = \mathcal{N}_\phi(\mathbf{z_r}|\boldsymbol{\mu}^r, \boldsymbol{\sigma}^r\mathbf{I}),$$
$$\boldsymbol{\mu}^r = \mathrm{MLP}^r_\phi(\mathbf{h_c}), \boldsymbol{\sigma}^r = \mathrm{softplus}(\mathrm{MLP}^r_\phi(\mathbf{h_c})),$$

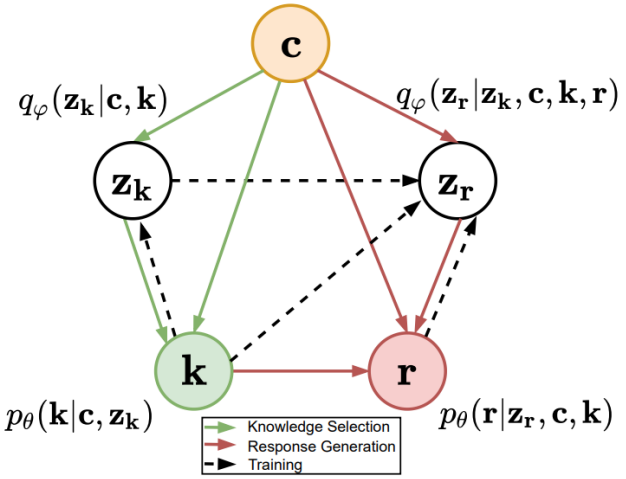


Figure 1: The graphical framework for CoLV model. **c**: dialogue context, **k**: knowledge, **r**: response. The dotted line denotes training procedure solely, while the solid line denotes both training and inference process.

=> Jointly end-to-end learn and Variational Inference!

Experiments

Model	WoW Test Seen						WoW Test Unseen					
	ACC	PPL	BLEU-4	RG-1	RG-2	Dist-2	ACC	PPL	BLEU-4	RG-1	RG-2	Dist-2
S2SA	-	93.85	0.46	12.53	0.69	4.81	-	120.81	0.34	9.30	0.76	11.53
Transformer	-	72.42	0.39	14.35	1.36	19.68	-	91.41	0.39	12.87	0.66	12.15
MemNet	21.60	63.52	0.41	16.9	0.64	24.16	13.82	96.47	0.32	14.46	0.82	16.27
PostKS	3.66	79.19	0.57	13.04	1.17	16.70	3.29	152.7	0.36	13.15	1.08	13.38
SKLS	26.83	52.09	1.35	16.87	6.84	23.13	16.59	81.44	1.05	16.16	4.21	16.42
DukeNet	25.96	48.33	2.46	19.02	6.54	25.67	17.49	69.38	1.68	19.36	5.23	17.03
PIPM	27.75	42.71	2.26	19.34	7.36	26.41	**19.43**	65.71	1.56	17.60	5.49	17.74
CoLV	**30.12**⋆	**39.56**⋆	**2.85**⋆	**20.62**	**7.89**	**29.74**⋆	18.91	**54.30**⋆	**2.12**⋆	**19.68**⋆	**6.31**	**20.13**⋆

Table 2: Automatic evaluation results on *WoW Test Seen* and *WoW Test Unseen* (%). The metrics Accuracy, Perplexity, ROUGE-1, ROUGE-2 and Distinct-2 are abbreviated as ACC, PPL, RG-1, RG-2 and Dist-2, respectively. The best results are highlighted with **bold**. "⋆" denotes that the result is statistically significant with $p < 0.01$.

	WoW	Holl-E
Training size	18,430	7,228
Validation size	1,948	930
Test size	965 (S)/968 (U)	913
Avg. Num of kg	67	53

Table 3: Statistics of two experimental datasets, *Wizard of Wikipedia* (WoW) and Holl-E. "**S**" and "**U**" denotes the test seen and test unseen in WoW dataset respectively.

Model	Holl-E					
	ACC	PPL	BLEU-4	RG-1	RG-2	Dist-2
S2SA	-	150.26	4.84	4.28	2.01	10.38
Transformer	-	120.31	5.09	6.72	2.96	14.29
MemNet	22.75	138.38	5.49	20.19	10.34	23.63
PostKS	1.56	187.20	5.85	15.23	6.08	19.74
SKLS	29.25	48.97	17.81	29.82	23.19	27.43
DukeNet	30.38	42.72	19.15	**32.64**	19.55	28.53
PIPM	30.67	39.22	18.27	30.81	23.96	27.20
CoLV	**32.65**⋆	**34.84**⋆	**20.33**⋆	31.97	**25.84**⋆	**29.86**⋆

Table 4: Automatic evaluation results on *Holl-E* (%). The best results are highlighted with **bold**. "⋆" denotes that the result is statistically significant with $p < 0.01$.

SOTA on WoW and Holl-E dataset with various metrics!!

Experiments

Dialogue Context: Have you been to Europe? → I have! I have been to British. → Great! I've only ever been to Canada.	
Knowledge Candidates: 1. Ireland is the third-largest island in Europe. Thick woodlands covered the island until the Middle Ages . 2. Ireland the second largest island in the British Isles, after Great Britain. 3....	
GT	That's pretty cool, but I'd still love to visit more of Europe. It's cool to explore woodland in Ireland.
SKLS	Yes, Ireland is the largest island in the Europe. I would like to go there again.
DukeNet	Canada is also a beautiful country, like the Ireland.
PIPM	Pretty cool! Canada is fantastic! I wish I lived there instead of the Europe.
Ours-1	That's great! Come to Ireland, the third largest island in Europe.
Ours-2	Pretty great! If you know that Ireland was covered by woodlands before many years.
Ours-3	That's pretty cool. The woodlands in Ireland deserves to be visited.
Ours-4	Yeah, Ireland has a third-largest woodlands. Have you ever heard it?

Table 8: Case study on the WoW Test Seen dataset. For fair comparison of the generation performance, we choose the case that all model select the same knowledge (marked as yellow). We further present several different responses that all generated by our model. "GT" denotes the ground truth.

Dialogue Context: Thierry Henry is one of my all time favorite players. What about you?	
Knowledge Candidates: 1. Thierry Daniel Henry is a retired French professional footballer . 2. He played as a forward and is the second assistant manager of the Belgium national team . 3. Henry made his professional debut with Monaco in 1994. 4. A year later he signed for Premier League club Arsenal .	
GT Pair	Kg: a retired French professional footballer. Response: He was good. he is a retired French professional footballer.
Pair-1	Kg: Premier League club Arsenal . Response: I know him. He has served in the premier league club Arsenal.
Pair-2	Kg: a retired French professional footballer . Response: Henry is a retired French footballer, he was so famous.
Pair-3	Kg: the second assistant manager of the Belgium national team . Response: Yes, I love him too. He was also the second assistant manager of the Belgium team.

Table 9: Qualitative analysis of collaborative latent variables. Knowledge-response pairs generated by our model. "GT pair" denotes the ground truth knowledge-response pair in the dataset. "Pair-1", "Pair-2" and "Pair-3" are generated from our model.

With jointly learned variables, various semantic response could be generated!!

Conclusion

- End-to-end jointly learn knowledge and response representation
- Reflect various relationship between knowledges and responses
- Variational inference for diverse semantic responses

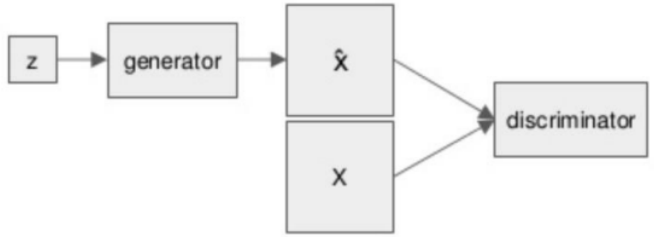
InterFaceGAN

Interpreting the Latent Space of GANs for Semantic Face Editing

CVPR 2020

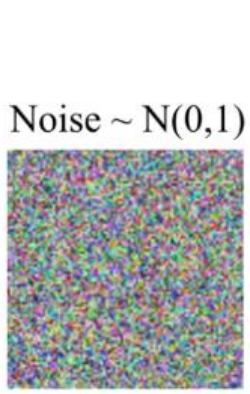
발표자: 김연아

Generative Adversarial Network



StyleGAN Generator

$g : \mathcal{Z} \rightarrow \mathcal{X}$



Gaussian Distribution

$\mathcal{Z}$

Z

latent vector Z

mapping

W

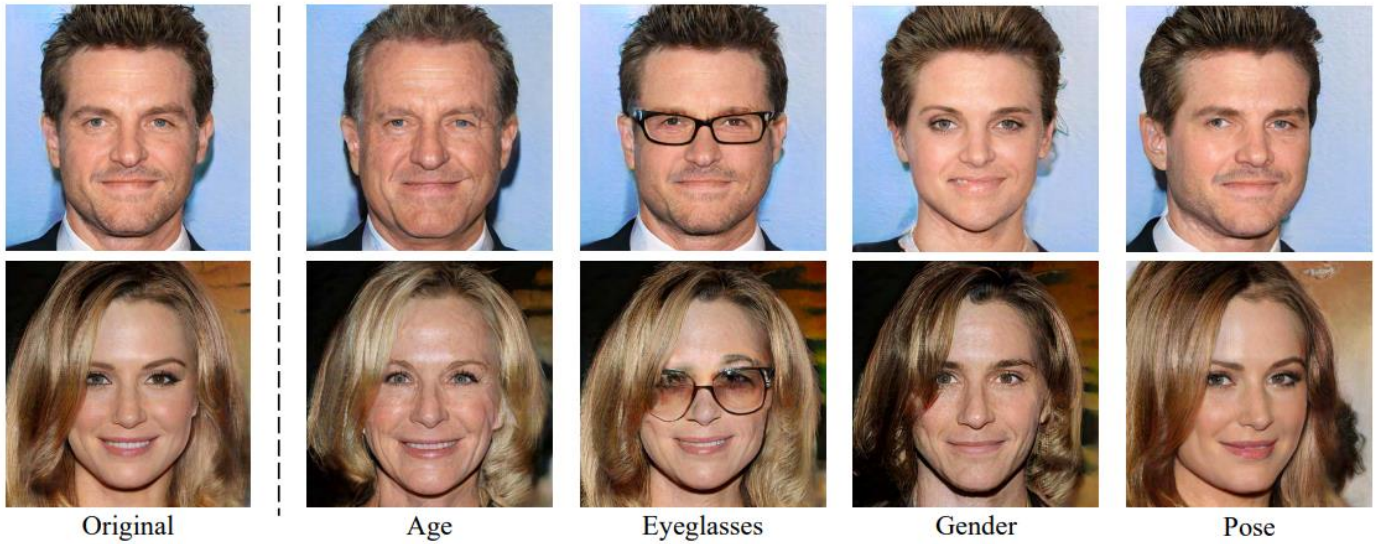
latent vector W

Synthesis Network

$\mathcal{X}$

Generated Image

Interpreting Face GANs



1. GAN의 latent space 내의 encoded 된 semantics 분석

2. Semantic image editing

Semantics in the latent space

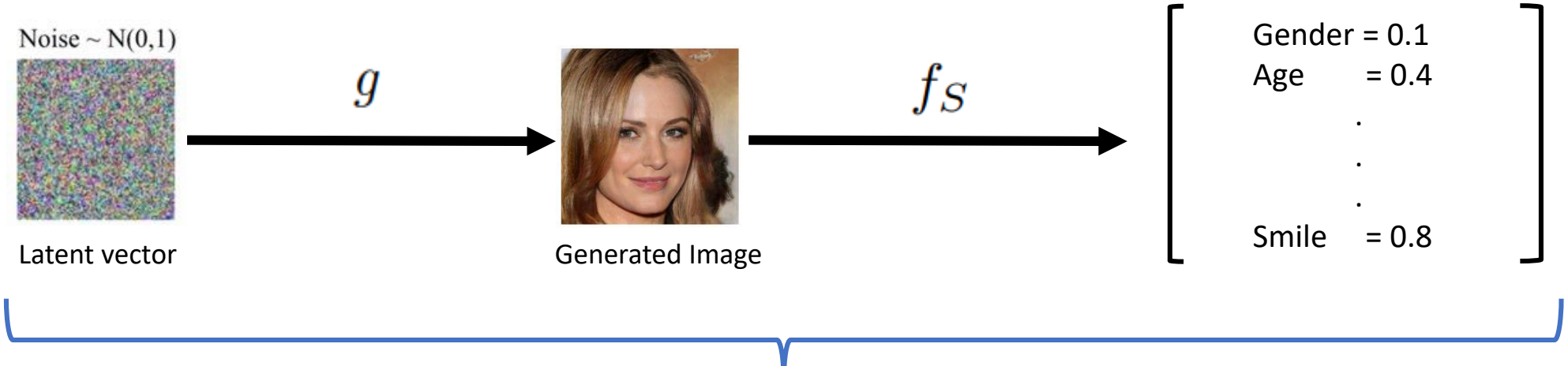
Generator

$$g : \mathcal{Z} \rightarrow \mathcal{X}$$

Semantic scoring function

$$f_S : \mathcal{X} \rightarrow \mathcal{S}$$

$\mathcal{Z}$ $\quad$ $\mathcal{X}$ $\quad$ $\mathcal{S} \subseteq \mathbb{R}^m$



$$\mathbf{s} = f_S(g(\mathbf{z}))$$

Train latent boundary (SVM)

Generator

$$g : \mathcal{Z} \rightarrow \mathcal{X}$$

$\mathcal{Z}$

Latent space

$\mathcal{X}$

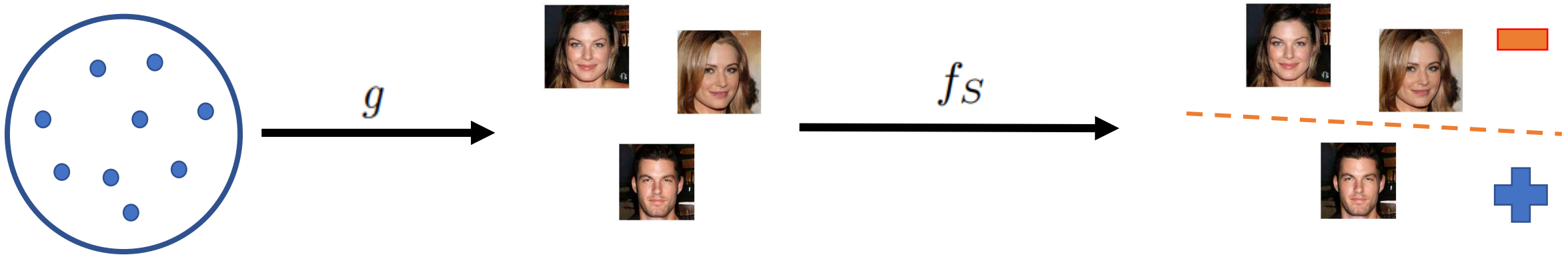
Generated Image Space

Semantic scoring function

$$f_S : \mathcal{X} \rightarrow \mathcal{S}$$

$\mathcal{S}$

Semantic Space



Train latent boundary (SVM)

Generator

$$g : \mathcal{Z} \rightarrow \mathcal{X}$$

Semantic scoring function

$$f_S : \mathcal{X} \rightarrow \mathcal{S}$$

$\mathcal{Z}$

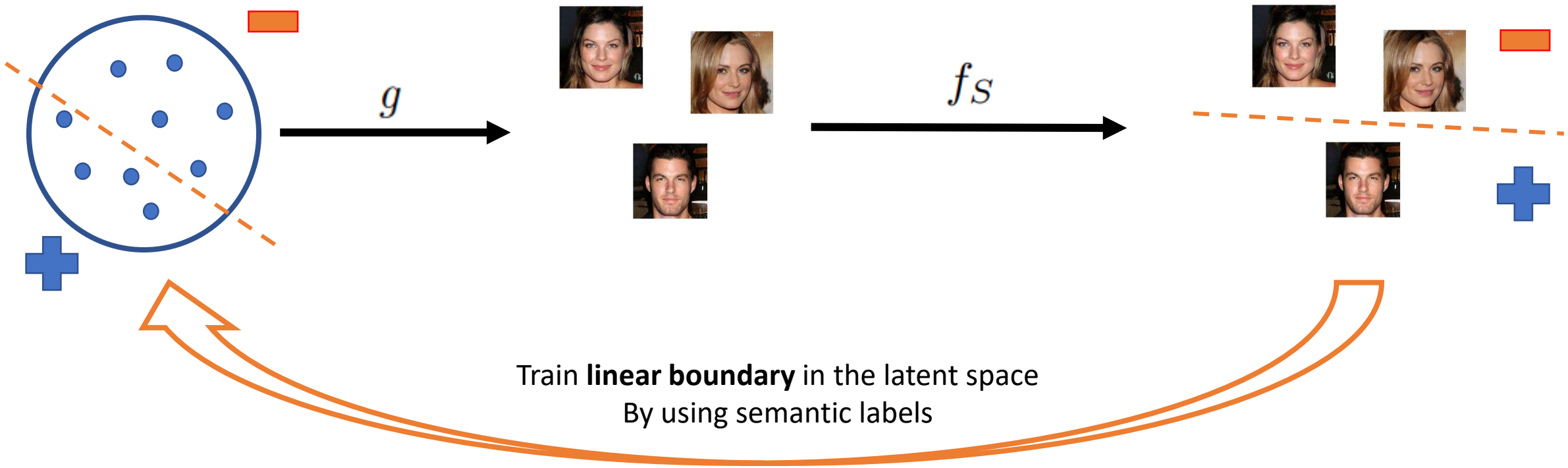
Latent space

$\mathcal{X}$

Generated Image Space

$\mathcal{S}$

Semantic Space

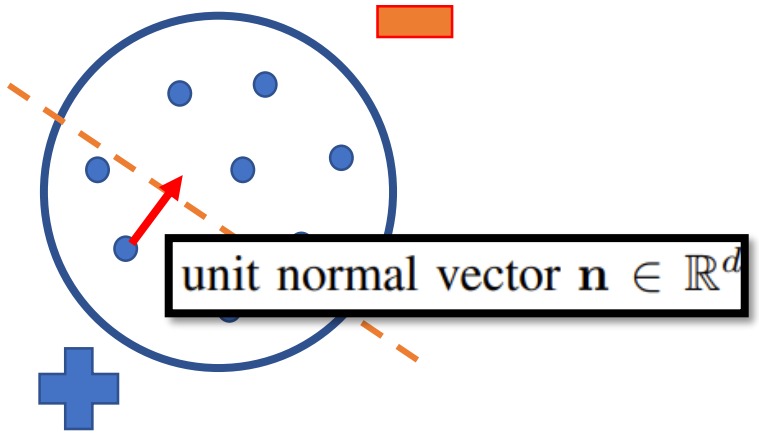


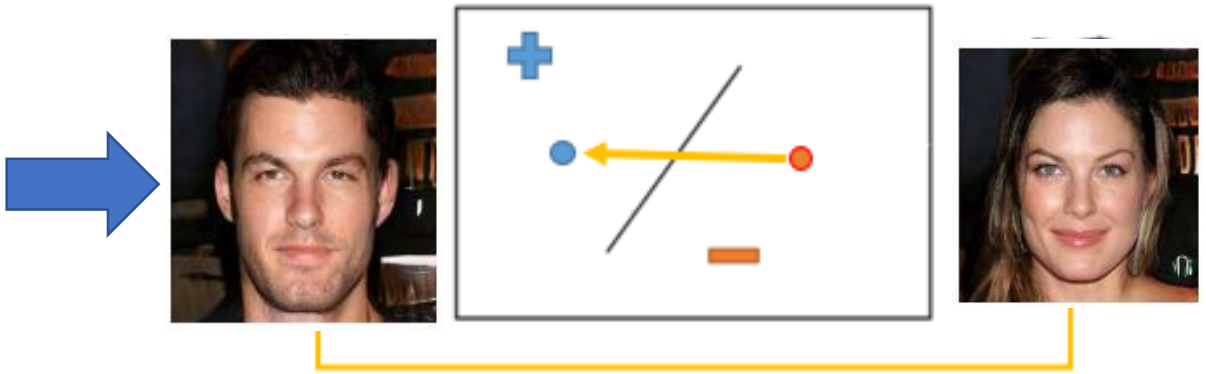
Train **linear boundary** in the latent space
By using semantic labels

Manipulation in the Latent Space

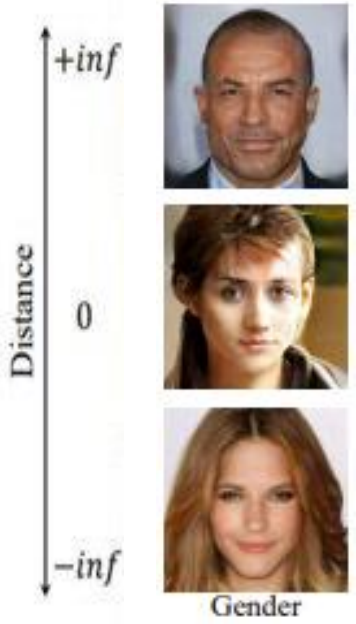
$\mathcal{Z}$

Latent space





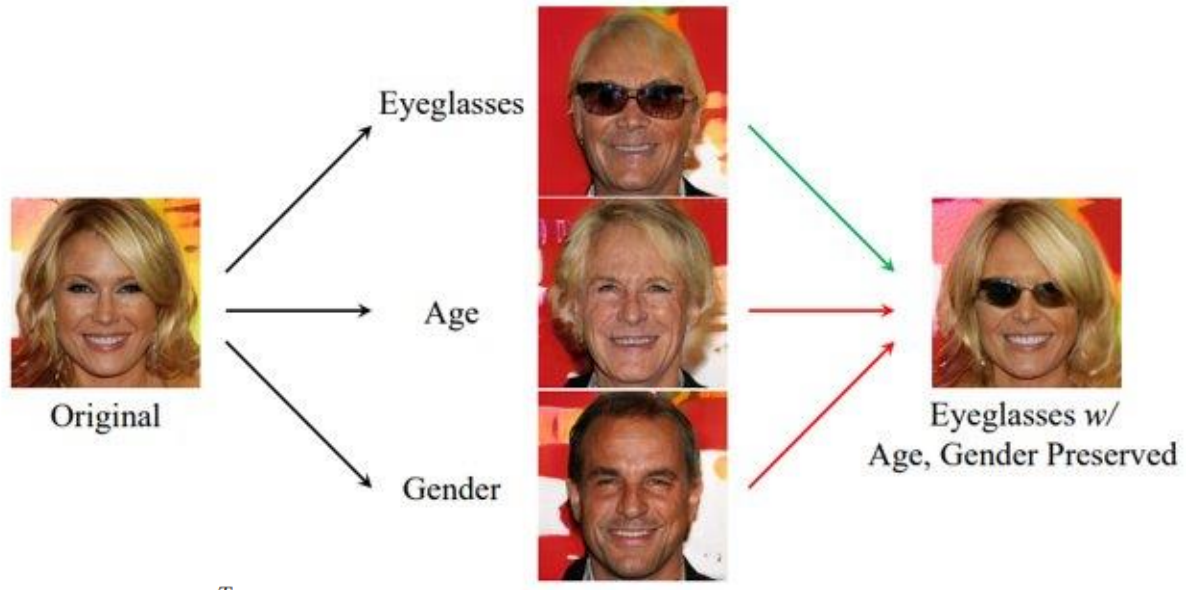
$$\mathbf{z}_{edit} = \mathbf{z} + \alpha \mathbf{n}$$

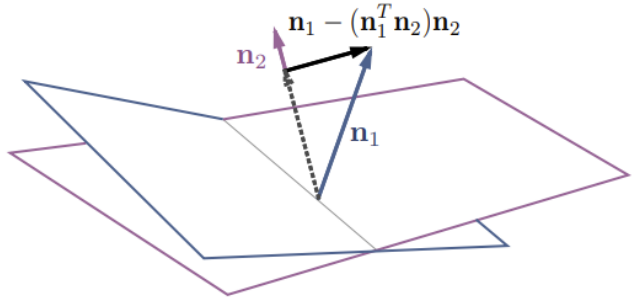


More results and explanation!

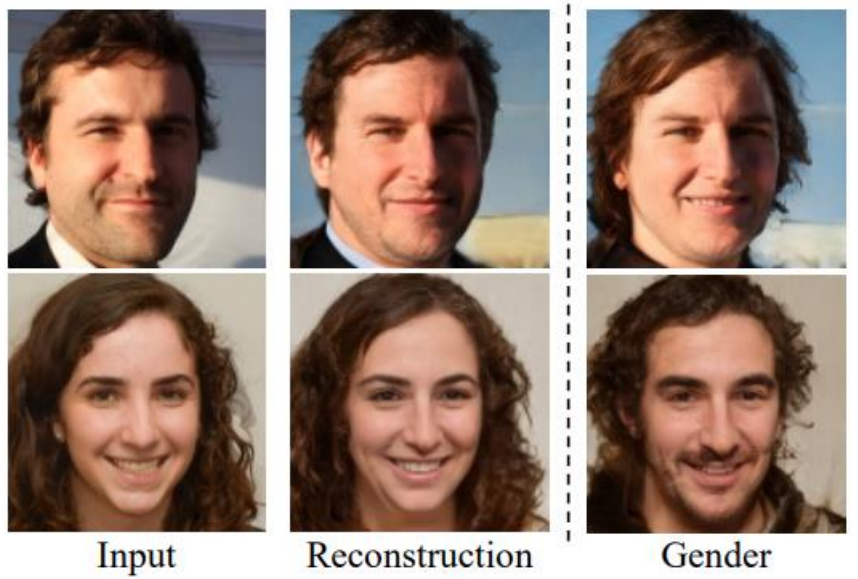
https://arxiv.org/pdf/1907.10786.pdf

Conditional Manipulation





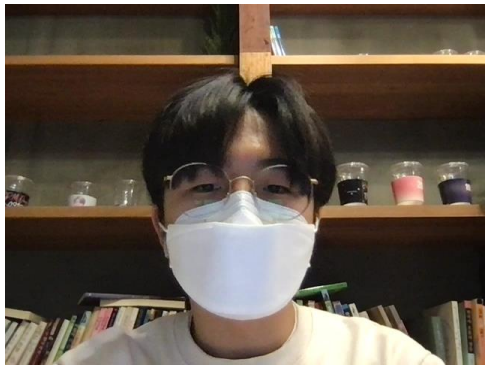
Real image editing



Long-short Distance Aggregation Networks for Positive Unlabeled Graph Learning (CIKM 19)

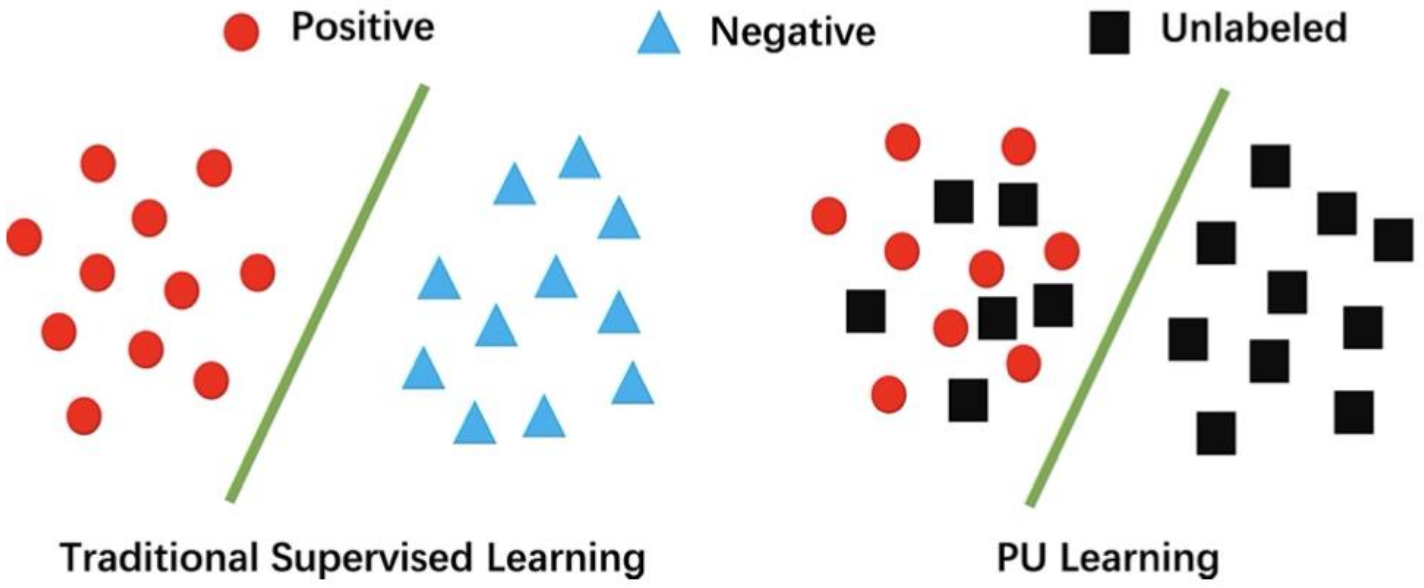
2020-22384

Junghun Kim



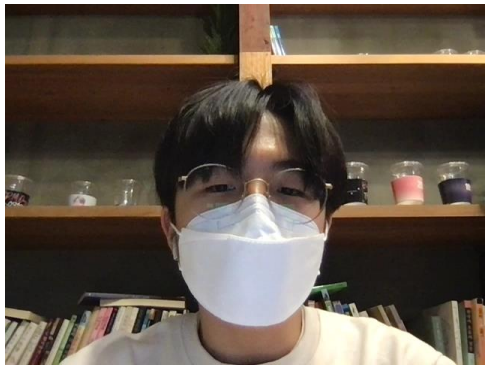
PU learning (1)

- **Positive-unlabeled (PU) learning**
  - Binary classification with limited observations
  - Negative examples are **unseen** during training
    - There are only positive and unlabeled examples



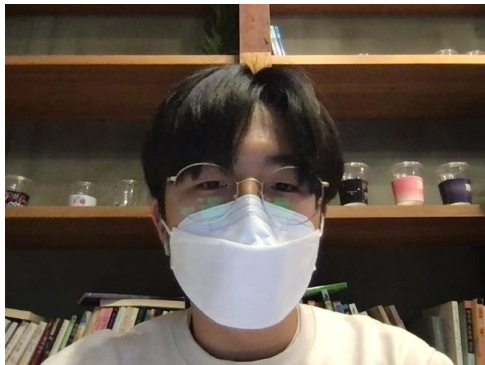
PU learning (2)

- PU learning is **common** in the real world
  - Detecting review manipulation
  - Detecting bot accounts in a social network
- Consider detecting review manipulation
  - We detected 100 reviews among 1000 ones
  - Are the remaining 900 reviews all normal?
  - They should be treated **unlabeled**, not **negative**



Problem Definition

- **Given**
  - Graph $G = (V, E)$
    - Set V consisting of positive node set P and unlabeled node set U
    - Set E of edges
- **Return**
  - A binary classifier model $f(G, P)$
- **Such that**
  - f classify a node into positive and negative
  - Detect negative nodes among the unlabeled ones



Proposed Approach

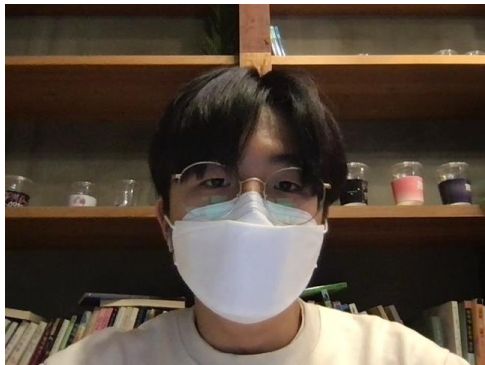
- **Short-Distance Attention**
  - One-hop self attention is used to learn a representation for each node
- **Long-short Distance Attention**
  - Compute self attention for $A^1, A^2, \ldots, A^K$
  - Aggregate representations obtained from $A^1, A^2, \ldots, A^K$
- **Unbiased PU Learning**
  - Non-negative risk estimator

$$\tilde{R}_{pu}(s) = \pi_p \hat{R}_p^+(s) + \max\left\{0, \hat{R}_u^-(s) - \pi_p \hat{R}_p^-(s)\right\}$$

$$\hat{R}_p^+(s) = (1/n_p) \sum_{i=1}^{n_p} \mathcal{L}(s(o_i^p), +1)$$

$$\hat{R}_p^-(s) = (1/n_p) \sum_{i=1}^{n_p} \mathcal{L}(s(o_i^p), -1)$$

$$\hat{R}_u^-(s) = (1/n_u) \sum_{i=1}^{n_u} \mathcal{L}(s(o_i^u), -1)$$



Experiments

- Proposed method outperforms all other competitors

Table 1: The F1 score on Citeseer.

%p	OC-SVM	Roc-SVM	F-PU	FS-PU	GCN	GAT	LSDAN
1	0.023	0.018	0.684	0.682	0.433	0.775	**0.786**
2	0.038	0.057	0.626	0.695	0.564	0.775	**0.804**
3	0.054	0.079	0.710	0.705	0.623	0.796	**0.813**
4	0.090	0.115	0.734	0.725	0.721	0.814	**0.828**

Table 2: The F1 score on DBLP.

%p	OC-SVM	Roc-SVM	F-PU	FS-PU	GCN	GAT	LSDAN
1	0.445	0.056	0.650	0.677	0.419	0.767	**0.808**
2	0.543	0.144	0.521	0.695	0.599	0.807	**0.833**
3	0.580	0.234	0.710	0.715	0.685	0.824	**0.824**
4	0.601	0.314	0.597	0.725	0.734	0.836	**0.849**

Pattern Recognition Final paper review
35th Conference on Neural Information Processing Systems (**NeurIPS 2021**)
Subject : Computer Vision and Pattern Recognition
Submitted on 15 Jun 2021

Revisiting the Calibration of Modern Neural Networks

Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, Mario Lucic

Google Research, Brain Team

김정현
ST ID : 2021-27679
junghyunkim@bi.snu.ac.kr
Biointelligence Laboratory
Seoul National Univertisy

Motivation

- Chapter 12 : Bayesian Decision
  - Risk formulation (Conditional Risk)
    - 암환자, 정상 example
    - $R(a_i|x) = \sum_{j=1}^{c} \lambda(a_i|w_j)p(w_j|x)$
    - Weighted Sum of Posterior probability($p(w_j|x)$)
    - Choose weights($\lambda(a_i|w_j)$) according to Posterior probability

    : Reliability of Posterior probability is important factor of choosing desirable $\lambda(a_i|w_j)$

    = Calibration

- About paper
  - Analysis on Calibration with image classification SOTA models

Calibration

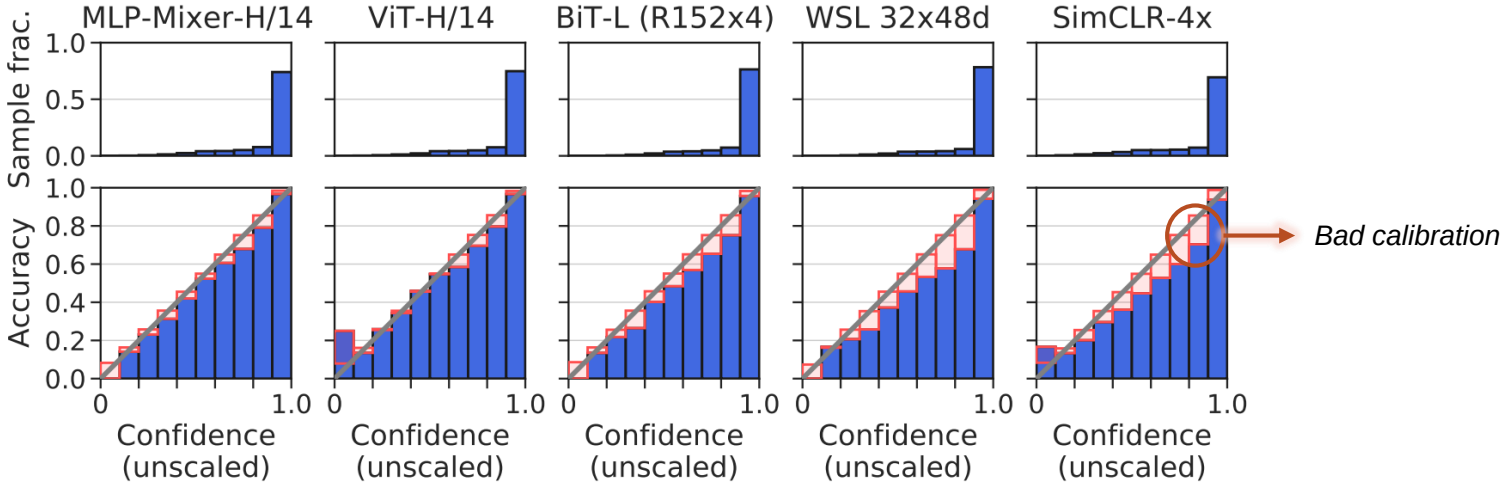
- Calibration
  - Discrepancy between Model confidence and Accuracy
  - e.g. If Expected Accuracy of samples that have Confidence of 80~90 is only 50%, Bad Calibration
- Measure1. Expected Calibration Error (ECE)
  - $ECE = \sum_{m=1}^{M} \frac{|B_M|}{n} |E(acc(B_m)) - E(conf(B_m))|$

$0 \le \hat{p}(x) < 0.2$	$0.2 \le \hat{p}(x) < 0.4$	$0.4 \le \hat{p}(x) < 0.6$	$0.6 \le \hat{p}(x) < 0.8$	$0.8 \le \hat{p}(x) < 1$
	$y_1 = \hat{f}(x_1)$ $y_2 \neq \hat{f}(x_2)$ $y_3 \neq \hat{f}(x_3)$	$y_4 = \hat{f}(x_4)$ $y_5 \neq \hat{f}(x_5)$ $y_6 = \hat{f}(x_6)$	$y_7 = \hat{f}(x_7)$ $y_8 \neq \hat{f}(x_8)$	$y_9 = \hat{f}(x_9)$ $y_{10} \neq \hat{f}(x_{10})$ $y_{11} = \hat{f}(x_{11})$
$E(acc(B_m))$	0.33	0.66	0.5	0.66
$E(conf(B_m))$	0.33	0.46	0.7	0.96
$Calibration\ Error$	0	0.2	0.2	0.3

$$ECE = \frac{3 * 0 + 3 * 0.2 + 2 * 0.2 + 3 * 0.3}{3 + 3 + 2 + 3}$$

Calibration

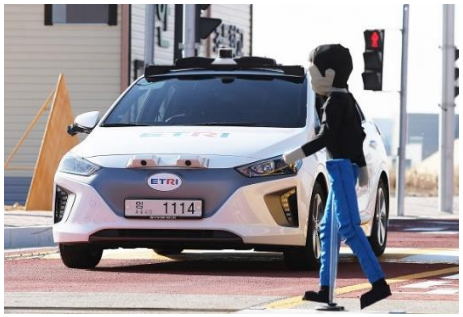
- Measure 2. Reliability Diagrams

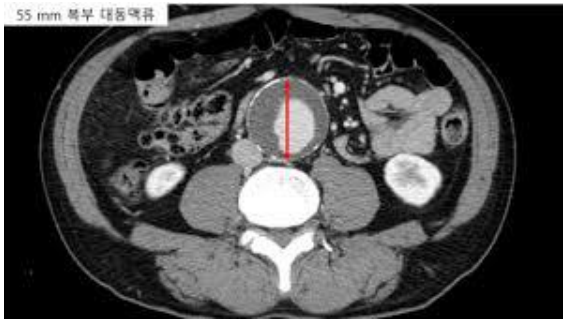


- Method 1. Temperature Scaling
  - $\widehat{q_i} = \max_k \sigma_{SM} \frac{z_i^{(k)}}{T}$
  - Increasing T asymptotically makes calibrated probability k^{-1} where k is number of class
  - No impact on model output but Calibration

Calibration

- Important measure in Safty-critical applications
  - Autonomous driving, Medical diagnosis, Forecasting

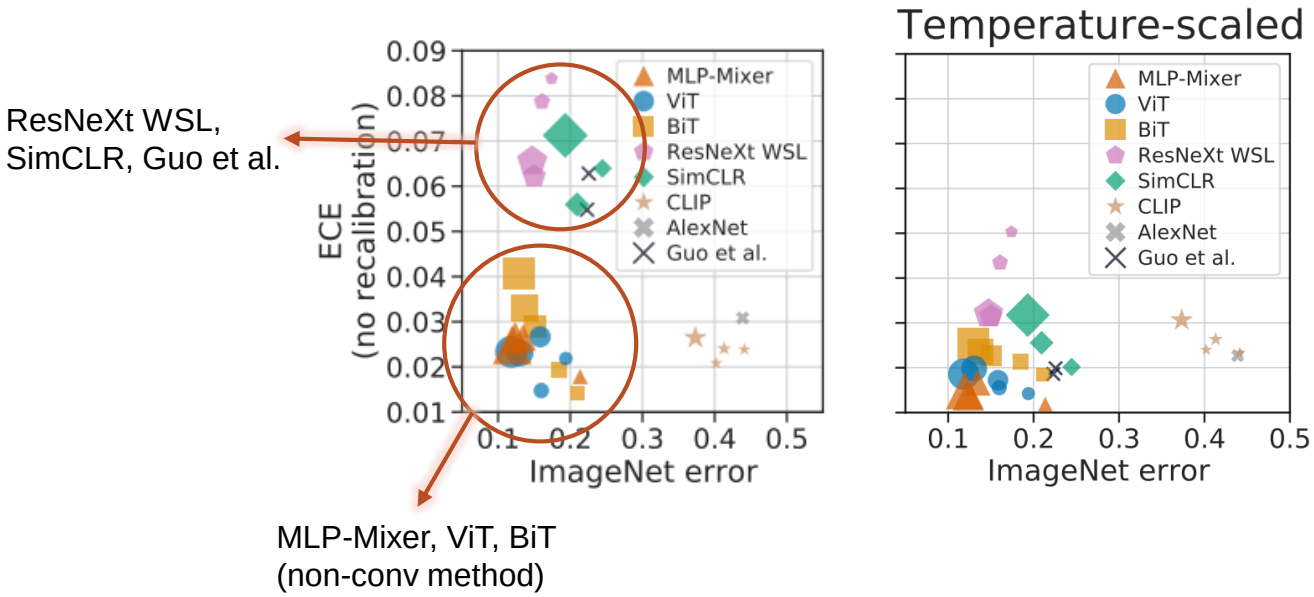




- Impact on Conditional Risk
  - $R(a_i|x) = \sum_{j=1}^{c} \lambda(a_i|w_j)p(w_j|x)$
  - $p(w_j|x)$ should be close an actual accuracy
    - e.g. acc$(\sim cancer|x) = 0.5$ but $p(\sim cancer|x) = 0.9$ then diagnose to normal regardless of λ
    - Importance of calibrated confidence i.e. discrepancy between posterior probability and actual Accuracy

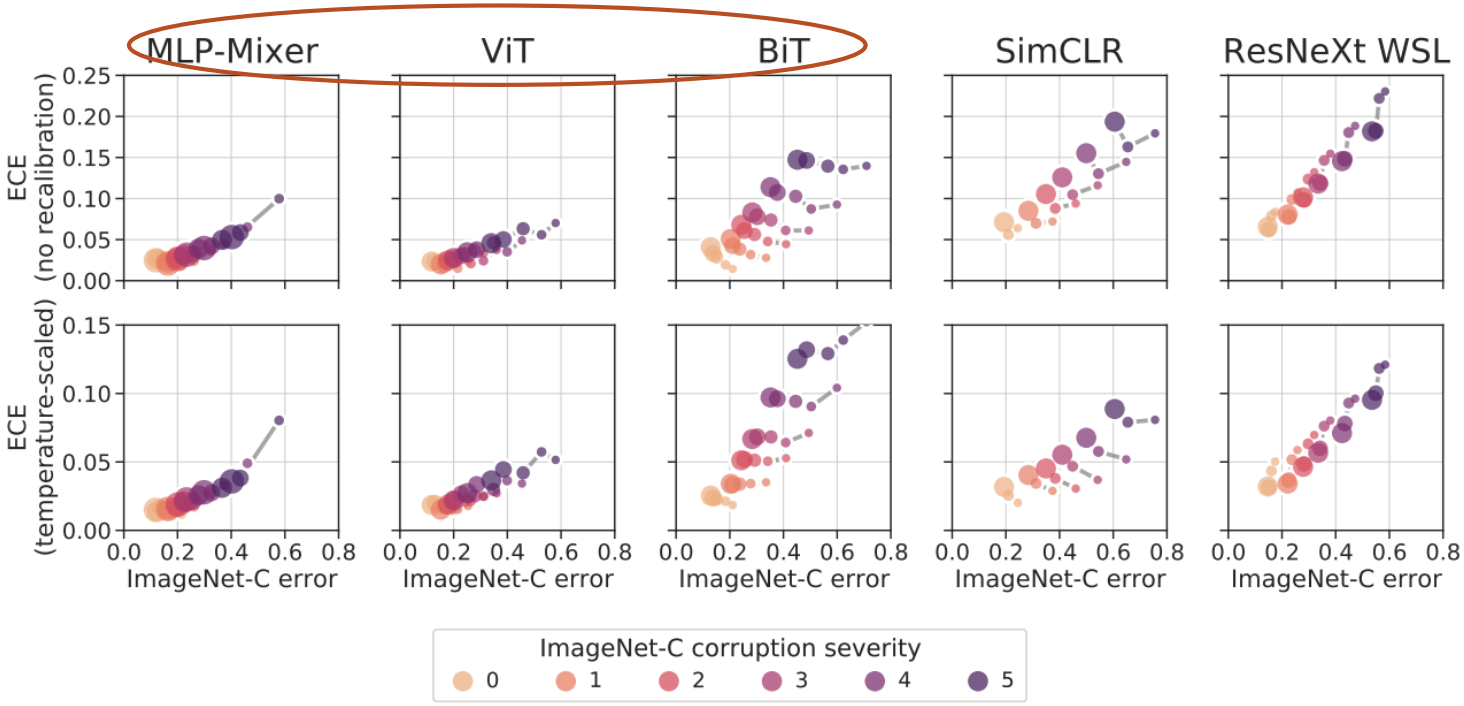
Analysis

- Calibration analysis on SOTA models
  - SOTA algorithms
    - MLP-Mixer, ViT, BiT, ResNeXt WSL, SimCLR, CLIP, Alexnet, Guo et al.
- Analysis (In distribution)
  1. Temperature scaling are effective on ECE
  2. ‘MLP-Mixer, ViT, BiT’(non convolution) better than ‘ResNeXt WSL, SimCLR, Guo et al.’ wrt ECE
  3. In most case, Model size ∝ Error ∝ 1/ECE
  4. Model-wise clustered output



Analysis

- Analysis (Distribution shift)
  1. (MLP-Mixer, ViT, BiT) better than (ResNeXt WSL, SimCLR, Guo et al.) wrt ECE, out-of-distribution. (MLP-Mixer, ViT, BiT) gives almost same result as in distribution ECE
  2. Corruption severity ∝ Error ∝ ECE



Corruption severity : degree of distribution shift

Conclusion

- More…
  - Different datasets
  - impact on size of dataset
  - training steps
  - relationship between Accuracy and ECE, …

- Comment
  - In safty-critical task using modern NN, refer to this paper to achieve better Calibration
  - Use ECE as well as Test error to justify your model

1. Revisiting the Calibration of Modern Neural Networks (NeurIPS 2021) https://arxiv.org/abs/2106.07998
2. On Calibration of Modern Neural Networks (ICML 2017) http://proceedings.mlr.press/v70/guo17a.html

SDE-Net: Equipping Deep Neural Networks with Uncertainty Estimates

Lingkai Kong [1] Jimeng Sun [2] Chao Zhang [1]

Jongwan Kim

Seoul National University
Data Science and AI Laboratory

Introduction

- DNNs have achieved enormous success in a wide spectrum of tasks
- DNNs are poor at quantifying uncertainties for their predictions
- Uncertainty quantification
  - Bayesian neural nets
    - often intractable
    - Specifying parameter priors for BNNs is challenging
  - Non-Bayesian approaches
    - Training can be prohibitively expensive in practice
    - Suffer from the drawback of conflating aleatoric uncertainty [1]

[1] Geifman, Yonatan, Guy Uziel, and Ran El-Yaniv. "Bias-reduced uncertainty estimation for deep neural classifiers." *arXiv preprint arXiv:1805.08206* (2018).

Contribution

- It explicitly models aleatoric uncertainty and epistemic uncertainty and is able to separate the two sources of uncertainties in its predictions
- It is efficient and straightforward to implement, avoiding the need of specifying model prior distributions and inferring posterior distributions as in BNNs
- It is applicable to both classification and regression tasks

Neural Net as Deterministic Dynamical System

- A neural network: $\mathrm{y} = \mathrm{f}(x)$ where $x: input, y: output$
- The transformations in ResNet can thus be viewed as the discretization of a dynamical system

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + f(\boldsymbol{x}_t, t).$$

$$\lim_{\Delta \to 0} \frac{\boldsymbol{x}_{t+\Delta t} - \boldsymbol{x}_t}{\Delta t} = \frac{d\boldsymbol{x}_t}{dt} = f(\boldsymbol{x}_t, t) \Longleftrightarrow d\boldsymbol{x}_t = f(\boldsymbol{x}_t, t)dt.$$

- The idea of the neural ODE method is to parameterize $f(x_t, t)$ with a neural net and exploit an ODE solver to evaluate the hidden unit state wherever necessary [2]

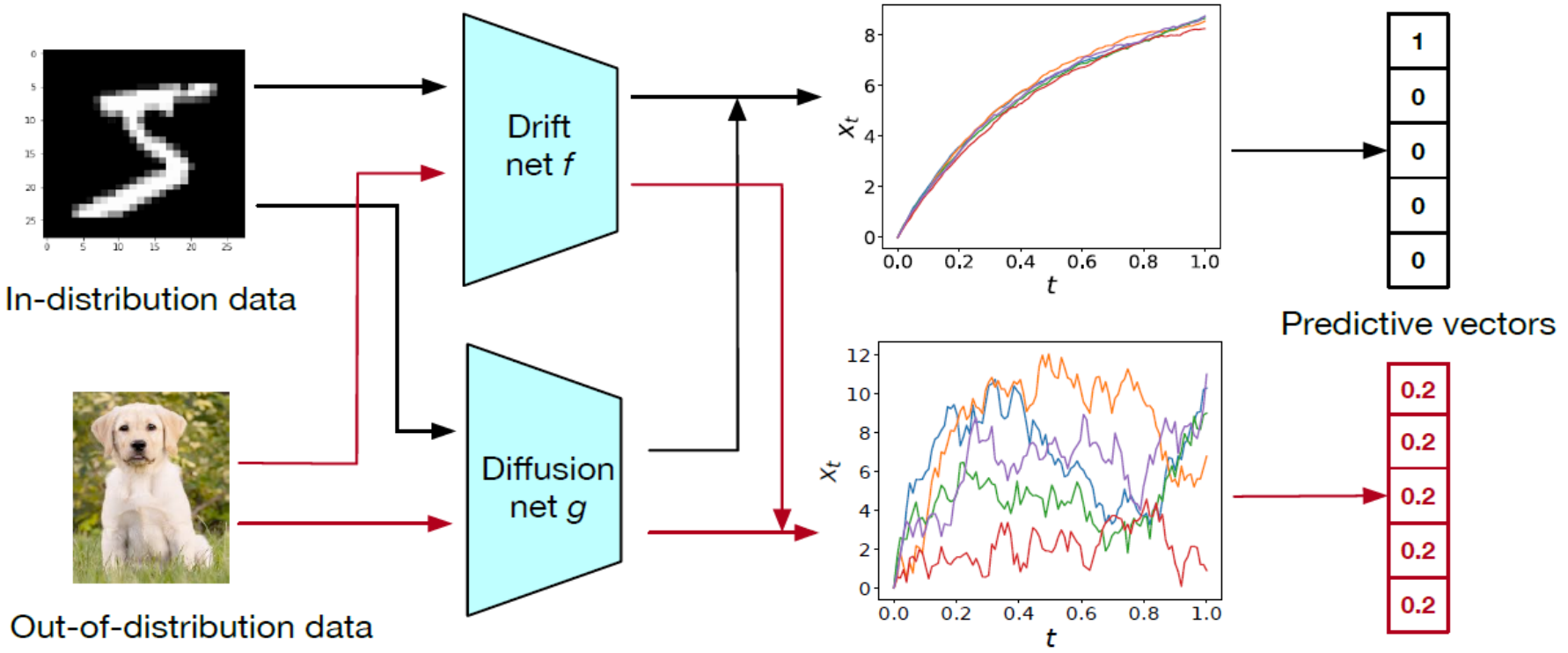
[2] Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. In Advances in Neural Information Processing Systems, pp.6571–6583, 2018

SDE-Net: Equipping Deep Neural Networks with Uncertainty Estimates

- ODE는 deterministic 하여 uncertainty를 추정하지 못함
  - Stochastic한 방법(SDE)을 사용

$$d\boldsymbol{x}_t = f(\boldsymbol{x}_t, t)dt + g(\boldsymbol{x}_t, t)dW_t.$$

Drift net　　Diffusion net



Trend

- (21 ICML) Neural SDEs as Infinite-Dimensional GANs
- (21 ICML) Continuous Latent Process Flows SDE
- (21 ICLR) learning continous time dynamics by SDE
- (21 ICCV) Neural TMDlayer Modeling Instantaneous flow of features via SDE Generators
- (21 ICLR_Best paper) Score-Based Generative Modeling through SDE


Thank You!

AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

Alexey Dosovitskiy[*,†], **Lucas Beyer**[*], **Alexander Kolesnikov**[*], **Dirk Weissenborn**[*], **Xiaohua Zhai**[*], **Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby**[*,†]

[*]equal technical contribution, [†]equal advising

Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com

Published as a conference paper at ICLR 2021

Reviewed by 김태규

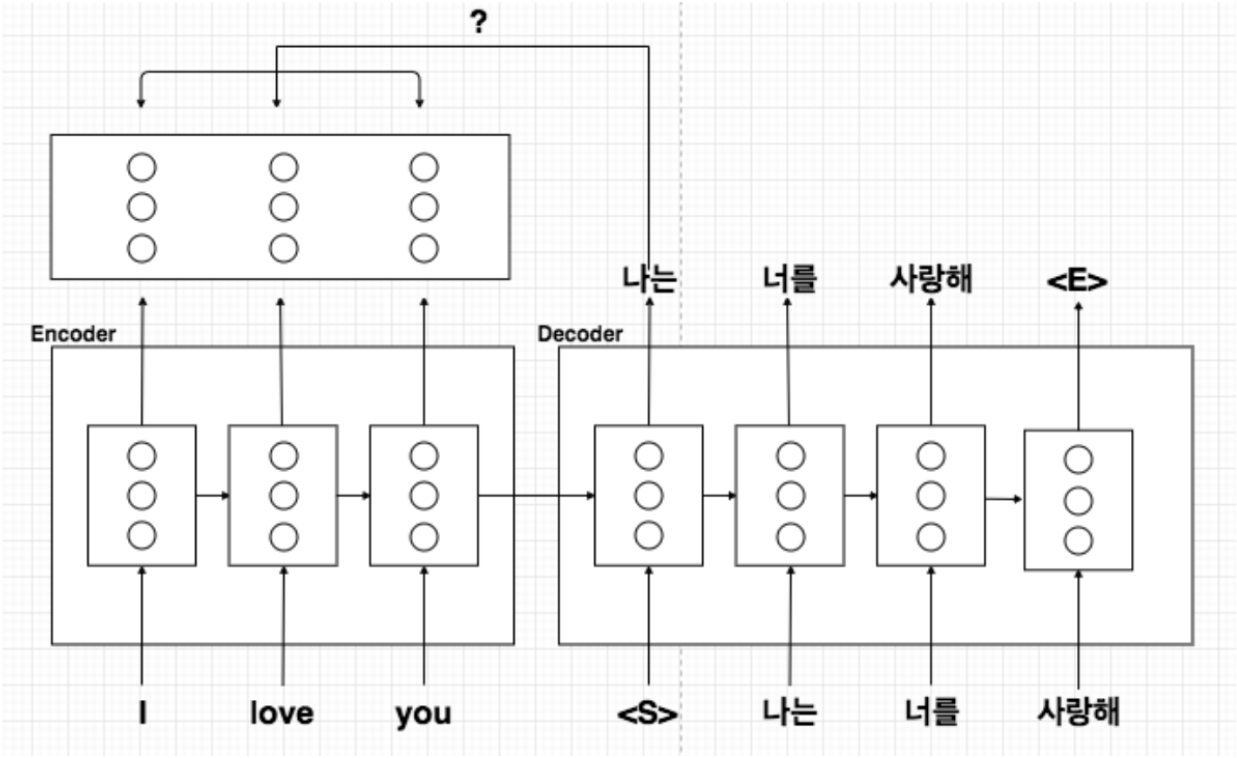
viT(Visual Transformer)?

- 현재까지 computer vision task는 cnn을 중심으로 진행
- Transformer 구조를 사용한 새로운 method 제시

Transformer?

- Attention을 먼저 알아야함

  -> Key Idea: Decoder의 특정 time-step의 output이 encoder의 모든 time-step의 output들 중, 어떤 output과 가장 연관이 있는가(Seq2seq에서 처음 제시된 개념)

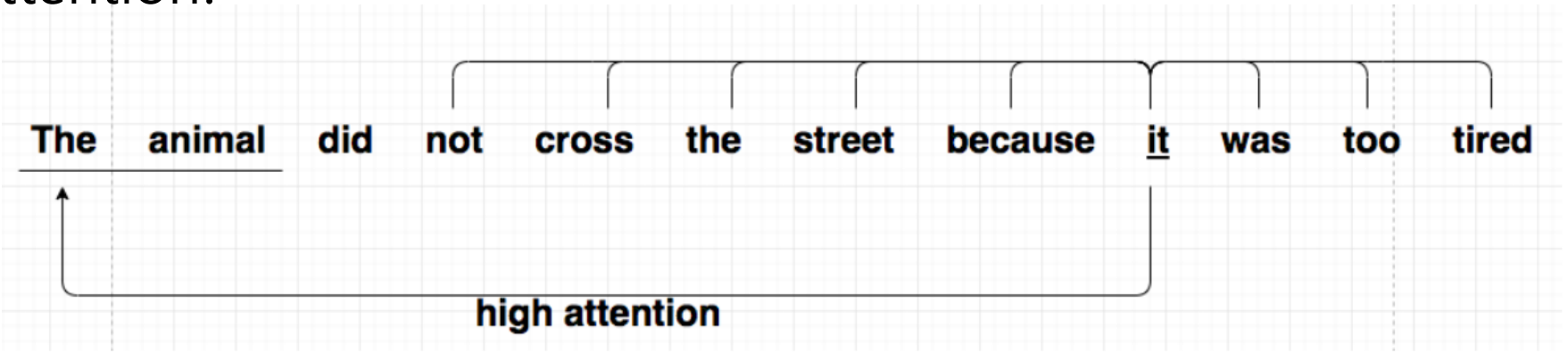


Transformer?

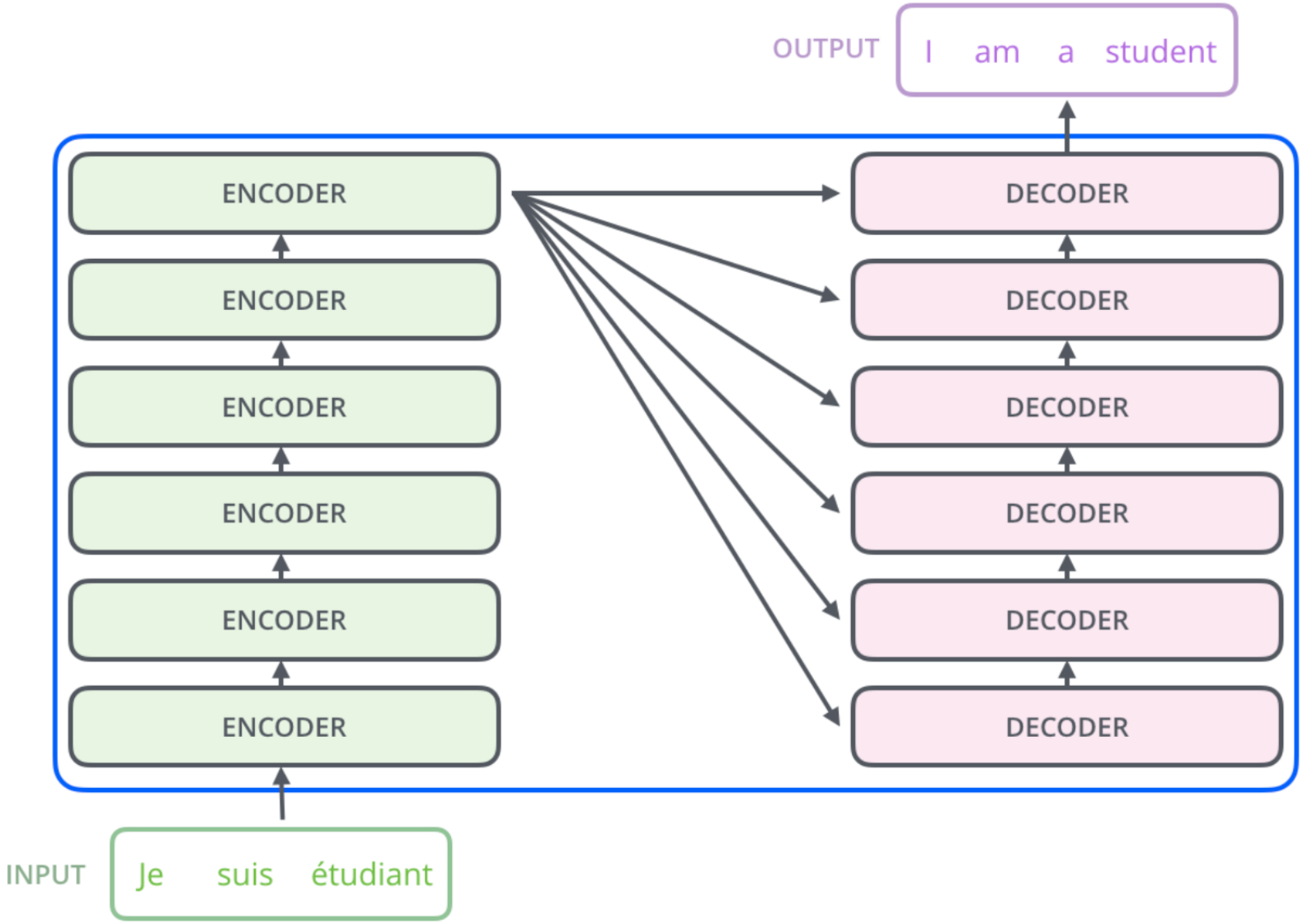
- RNN, LSTM 등을 사용하지 않고, '**attention**'을 이용해서 그 자체만으로 time sequence 역할을 할 수 있게 하는 모델

- 기존에 attention을 사용한 seq2seq 모델은, source language와 target language 간의 관계는 어느정도 알 수 있지만, 자신의 언어에 대해서는 그 관계를 알기가 힘듦

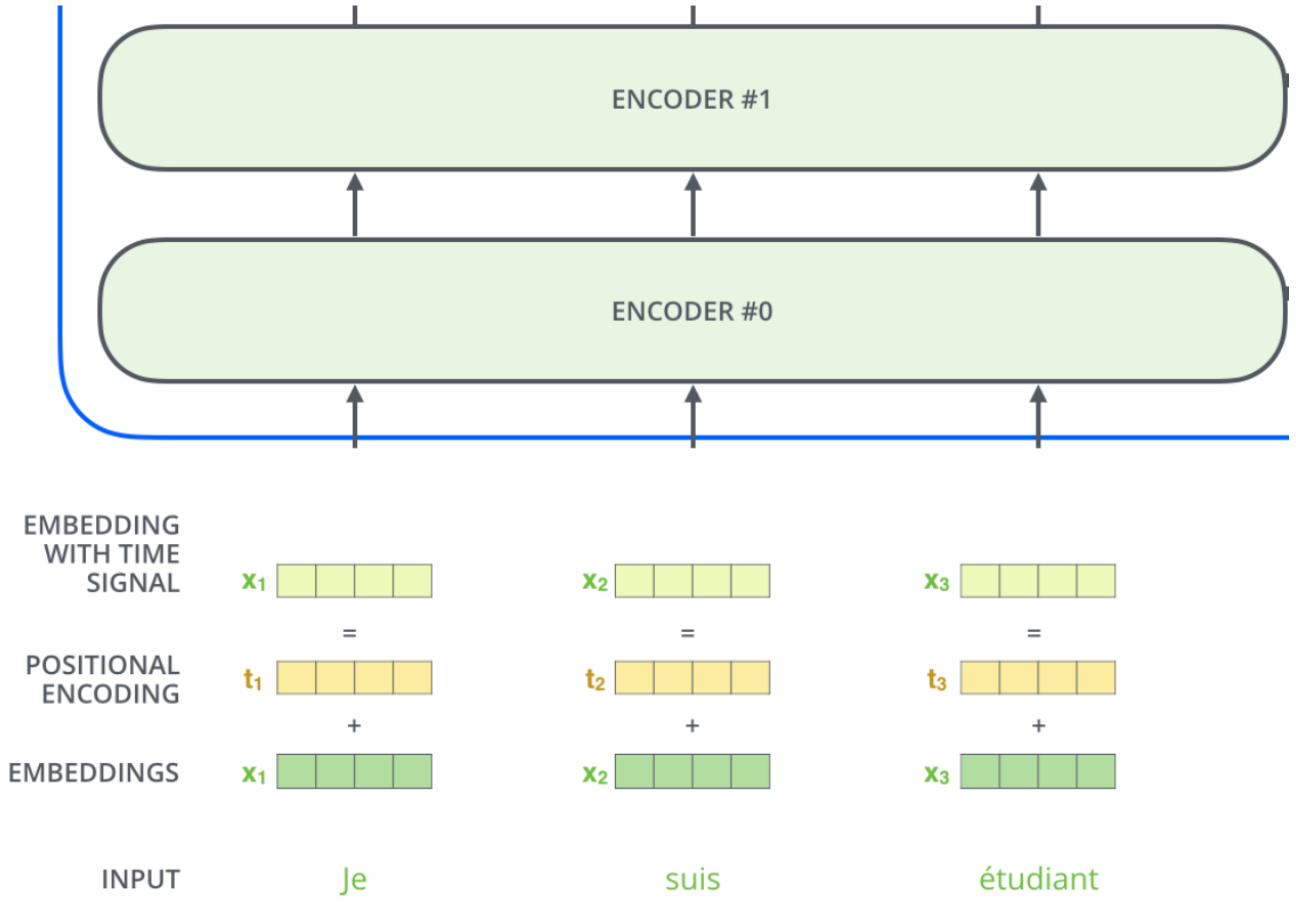
  -> self-attention!



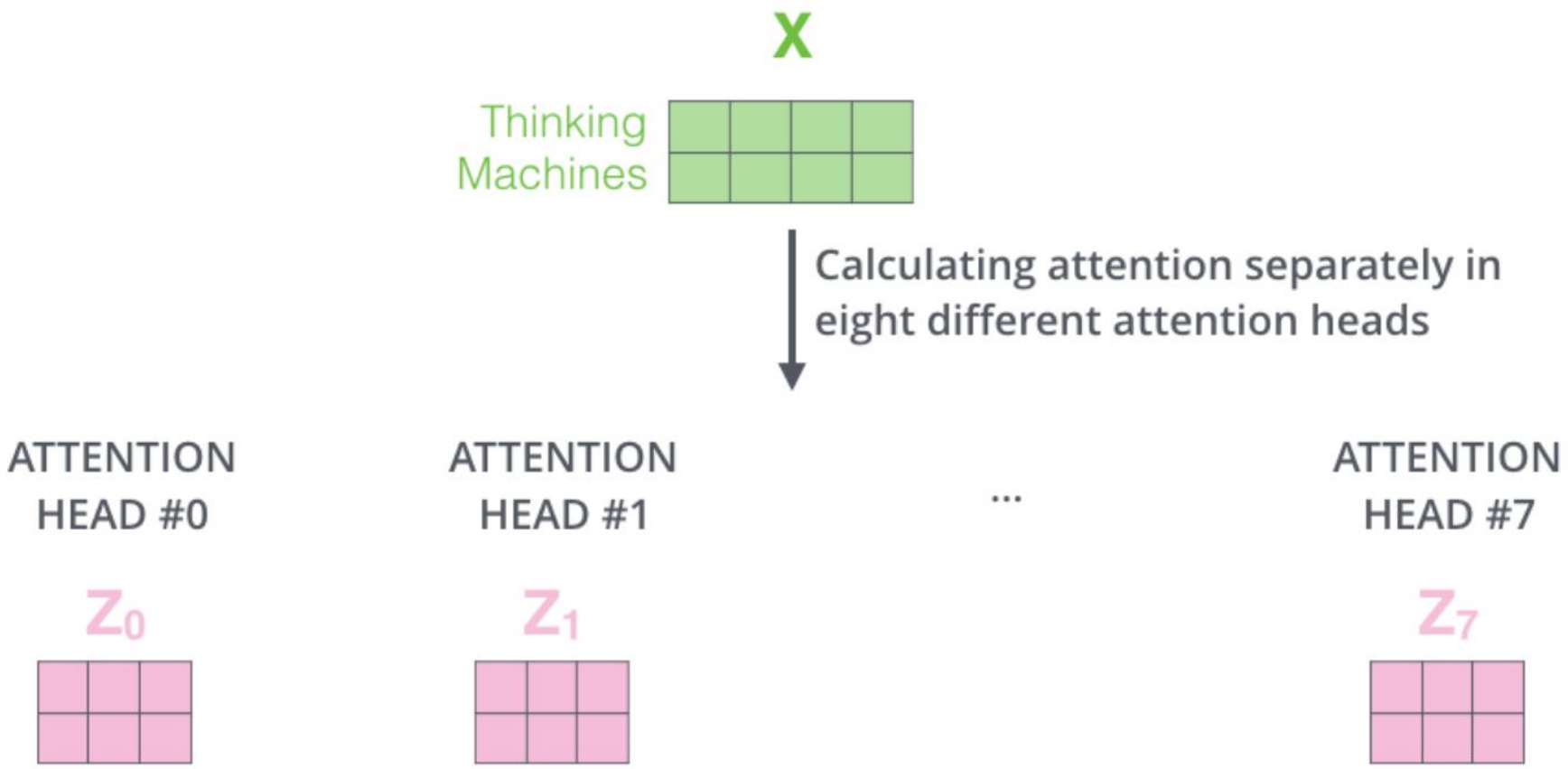
Transformer? - Architecture



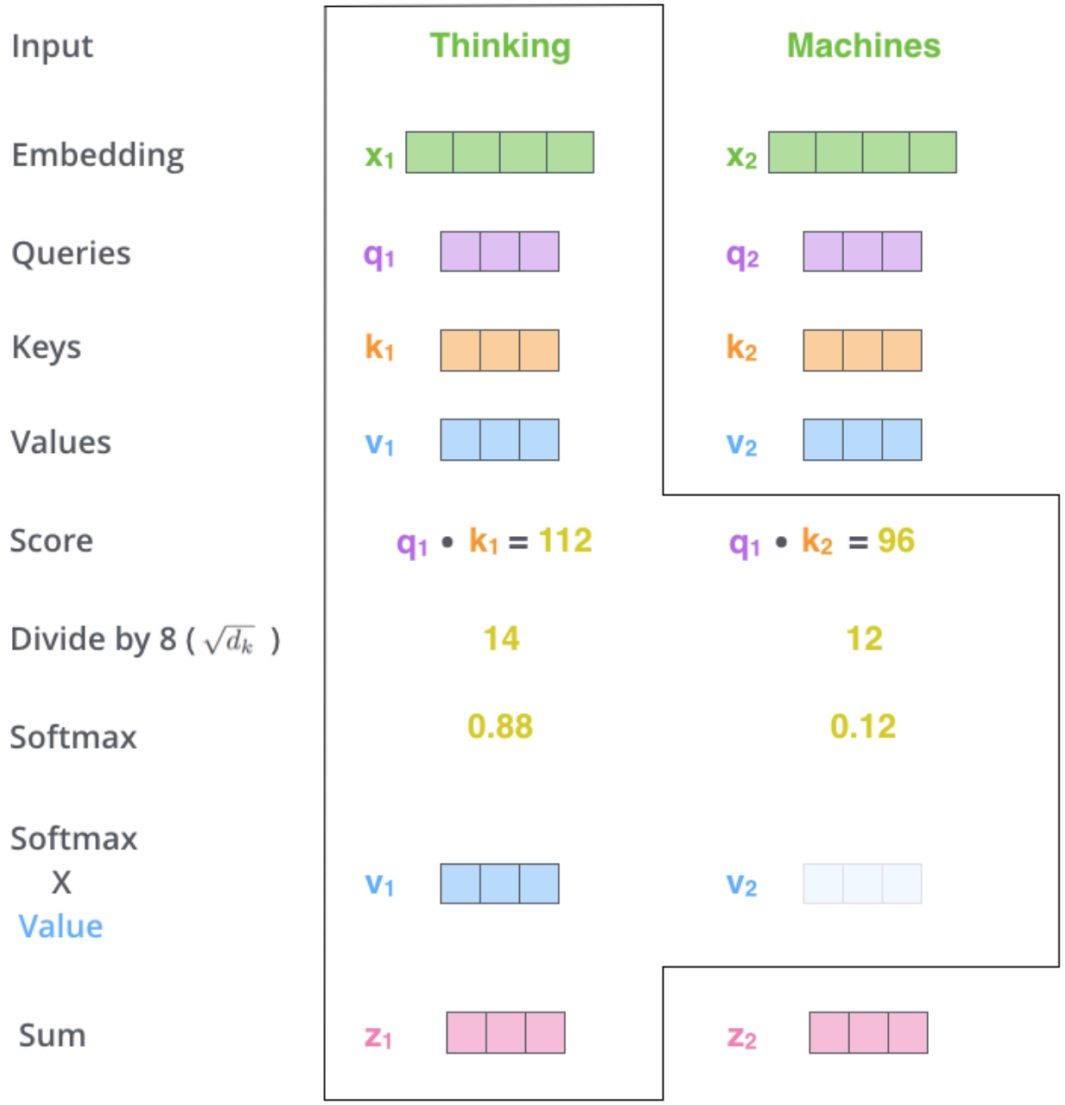
Transformer? – positional encoding



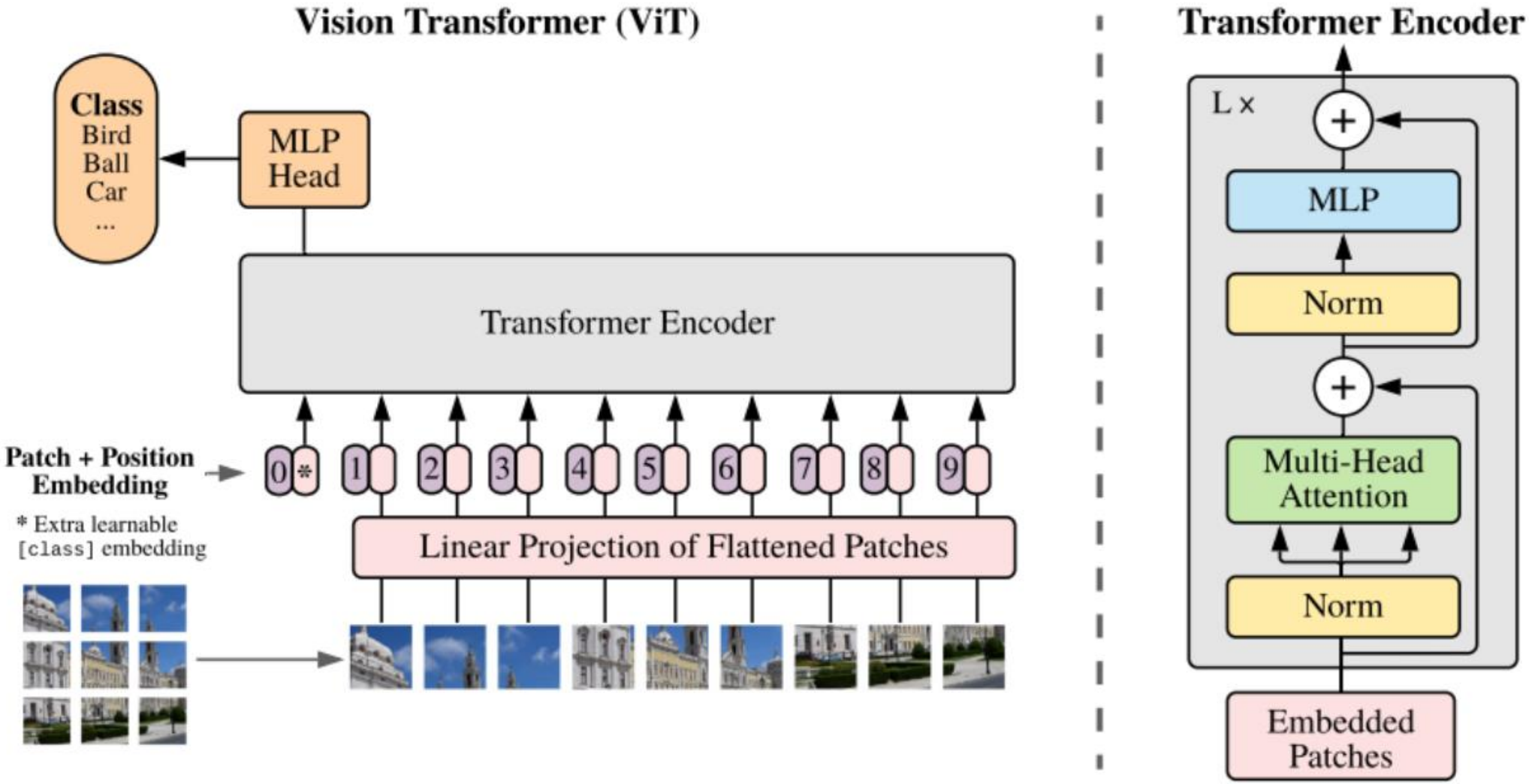
Transformer? – Multi-head attention



Transformer? – self attention



viT? - Architecture



Experiments & Conclusion

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

- Pre-trained 한 dataset의 크기가 클수록 ViT의 정확도 높음
  -> CNN과 다르게 convolutional inductive bias(locality, translation equivariance)가 없기 때문

- Dataset을 키울수록 성능이 좋아질 것(상대적인 pre-trained 비용도 작음)

패턴인식 논문 발표

2021.11.20

Heewon Kim

CONTRASTIVE REPRESENTATION DISTILLATION

Yonglong Tian
MIT CSAIL
yonglong@mit.edu

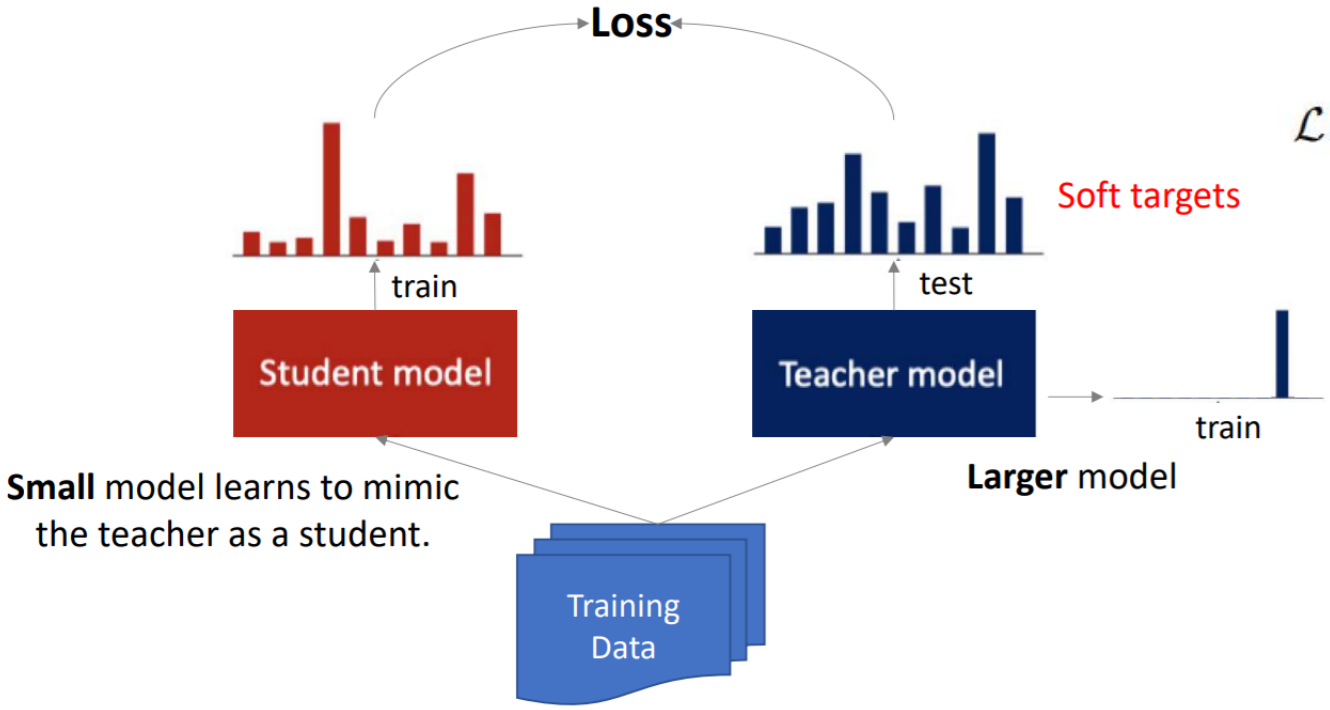
Dilip Krishnan
Google Research
dilipkay@google.com

Phillip Isola
MIT CSAIL
phillipi@mit.edu

Knowledge Distillation

- **Original KD[1]**
  - 미리 잘 학습된 **Teacher network**의 지식을 실제로 사용하고자 하는 **Student network**에게 전달하는 것이 목적.
  - **Teacher network**로부터의 **soft**한 **output** 확률 분포를 **Student**가 따라가도록 학습

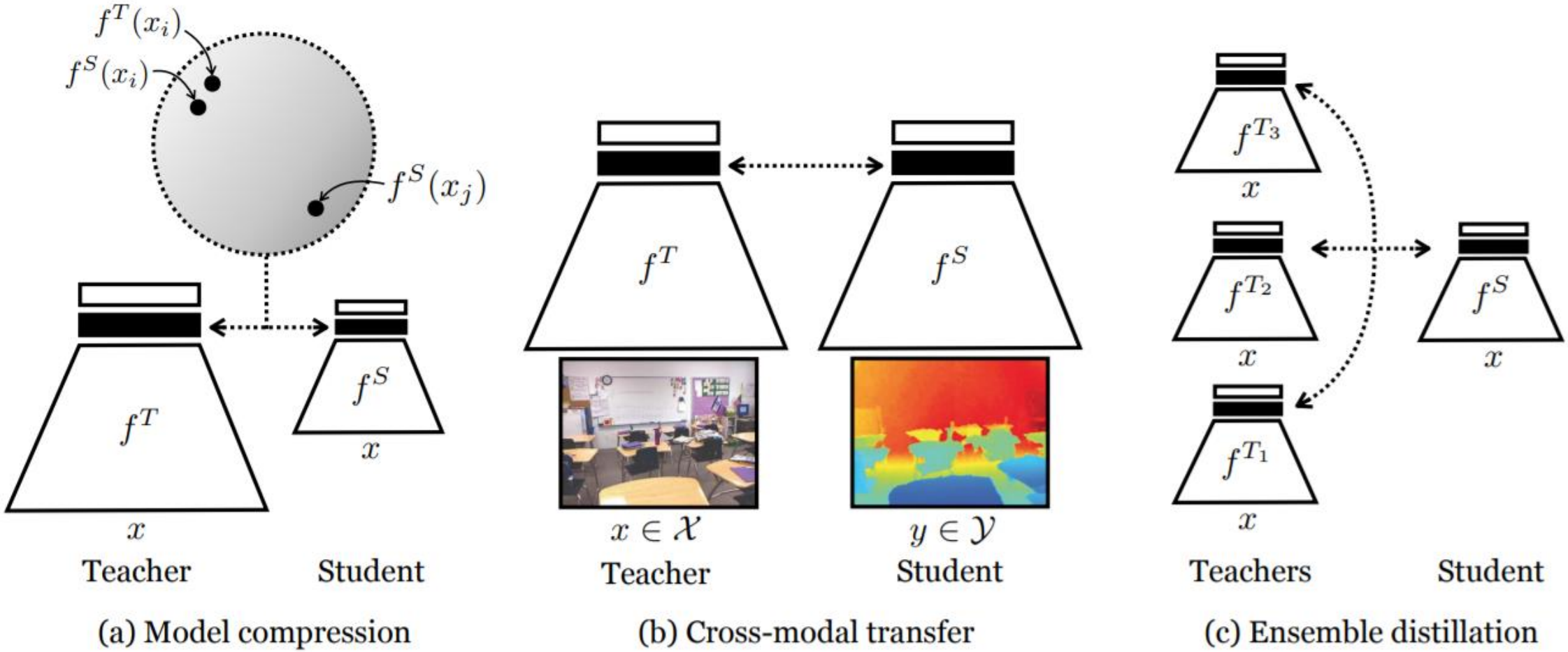
► **Knowledge**를 충분히 전파하지 못함. **Ignores Structural knowledge!**



$$\mathcal{L} = (1-\alpha)\mathcal{L}_{\text{CE}}(\mathbf{y}, \sigma(\mathbf{z}_S)) + 2\alpha T^2 \mathcal{L}_{\text{CE}}(\sigma(\frac{\mathbf{z}_S}{T}), \sigma(\frac{\mathbf{z}_T}{T}))$$

1) Geoffrey Hinton, Oriol Vinyals, and Jeff Dean.. Distilling the Knowledge in a Neural Network. NeurIPS 2014

Representation Distillation



1) Geoffrey Hinton, Oriol Vinyals, and Jeff Dean.. Distilling the Knowledge in a Neural Network. NeurIPS 2014

Method

Notation

$x \sim p_{\texttt{data}}(x)$ ◁ **data**

$S = f^S(x)$ ◁ **student's representation**

$T = f^T(x)$ ◁ **teacher's representation**

목표: T와 S의 Mutual Information $I(T;S)$을 최대화!

Objective Function:

$$f^{S*} = \arg\max_{f^S} \max_h \mathcal{L}_{critic}(h) \quad \triangleleft \quad \textbf{our final learning problem}$$

$$= \arg\max_{f^S} \max_h \mathbb{E}_{q(T,S|C=1)}[\log h(T,S)] + N\mathbb{E}_{q(T,S|C=0)}[\log(1-h(T,S))]$$

$$h : \{\mathcal{T}, \mathcal{S}\} \rightarrow [0,1]. \qquad h(T,S) = \frac{e^{g^T(T)' g^S(S)/\tau}}{e^{g^T(T)' g^S(S)/\tau} + \frac{N}{M}}$$

Mutual Information Lower Bound

Consider joint distribution $p(T,S)$ and product of marginal distribution $p(T)p(S)$.
Define distribution q and latent varible C.

$$q(T,S|C=1) = p(T,S), \quad q(T,S|C=0) = p(T)p(S)$$
$$q(C=1) = \frac{1}{N+1}, \quad q(C=0) = \frac{N}{N+1}$$

Posterior for class $C=1$

$$q(C=1|T,S) = \frac{q(T,S|C=1)q(C=1)}{q(T,S|C=0)q(C=0) + q(T,S|C=1)q(C=1)} = \frac{p(T,S)}{p(T,S) + Np(T)p(S)}$$

Here, We can observe a connection to mutual information:

$$\begin{aligned}\log q(C=1|T,S) &= \log \frac{p(T,S)}{p(T,S) + Np(T)p(S)} \\ &= -\log(1 + N\frac{p(T)p(S)}{p(T,S)}) \leq -\log(N) + \log \frac{p(T,S)}{p(T)p(S)}\end{aligned}$$

참고:

$$I(T;S) = \mathbb{E}_{P(T,S)} \log \frac{P(T,S)}{P(T)P(S)}$$

Then taking expection on both sides w.r.t. $p(T,S)$ (=$q(T,S|C=1)$)

$$I(T;S) \geq \log(N) + \mathbb{E}_{q(T,S|C=1)} \log q(C=1|T,S) \quad \triangleleft \quad \textbf{MI bound}$$

Maximize MI bound

Since we do not know the true distribution $q(C = 1|T, S)$,
We estimate it by fitting a model h: $\{T, S\} \rightarrow [0, 1]$.

$$\mathcal{L}_{critic}(h) = \mathbb{E}_{q(T,S|C=1)}[\log h(T, S)] + N\mathbb{E}_{q(T,S|C=0)}[1 - \log(h(T, S))]$$

$$h^* = \arg\max_h \mathcal{L}_{critic}(h) \quad \triangleleft \quad \textbf{optimal critic}$$

$h^*(T, S) = q(C = 1|T, S)$
See Appendix!

$$I(T; S) \geq \log(N) + \mathbb{E}_{q(T,S|C=1)} \log q(C = 1|T, S) \quad \triangleleft \quad \textbf{MI bound}$$

$$= \log(N) + \mathbb{E}_{q(T,S|C=1)}[\log h^*(T, S)] \quad ①$$

$$\geq \log(N) + \mathbb{E}_{q(T,S|C=1)}[\log h^*(T, S)] + N\mathbb{E}_{q(T,S|C=0)}[\log(1 - h^*(T, S))] \quad ②$$

$$= \log(N) + \mathcal{L}_{critic}(h^*) = \log(N) + \max_h \mathcal{L}_{critic}(h) \quad ③$$

$$\geq \log(N) + \mathcal{L}_{critic}(h) \quad ④$$

$$f^{S*} = \arg\max_{f^S} \max_h \mathcal{L}_{critic}(h)$$

Final Learning Objective

$$f^{S*} = \arg\max_{f^S} \mathcal{L}_{critic}(h)$$

$$h : \{\mathcal{T}, \mathcal{S}\} \rightarrow [0, 1].$$

$$h(T, S) = \frac{e^{g^T(T)' g^S(S)/\tau}}{e^{g^T(T)' g^S(S)/\tau} + \frac{N}{M}}$$

Thank you

Don't Stop Pretrainig:
Adapt Language Models to Domains and Tasks

Pattern Recognition 류정현

OUTLINE

1. Introduction
2. Domain-Adaptive Pretraining (DAPT)
3. Task-Adaptive Pretraining (TAPT)
4. Augmenting Training Data for Task-Adaptive Pretraining

INTRODUCTION

Do the latest large pretrained models work universally or is it still helpful to build separate pretrained models for specific domains?

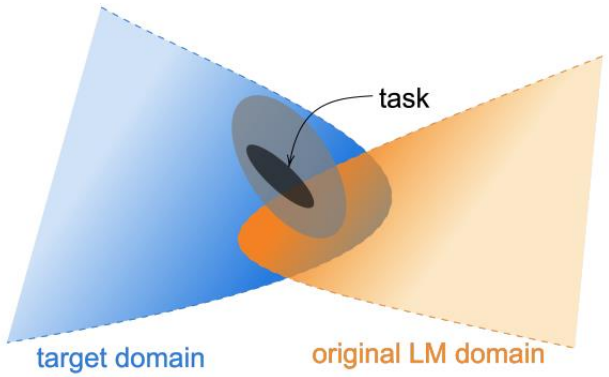


Figure 1: An illustration of data distributions. Task data is comprised of an observable task distribution, usually non-randomly sampled from a wider distribution (light grey ellipsis) within an even larger target domain, which is not necessarily one of the domains included in the original LM pretraining domain – though overlap is possible. We explore the benefits of continued pretraining on data from the task distribution and the domain distribution.

Domain-Adaptive Pretraining (DAPT)
: continue pretraining RoBERTa on a large corpus of unlabeled domain-specific text

Task-Adaptive Pretraining (TAPT)
: pretraining on the unlabeled training set for a given task
- smaller pretraining corpus, but much more task relevant

Experiment base setting : 4 domain & 8 tasks, RoBERTa

Domain	Task	Label Type	Train (Lab.)	Train (Unl.)	Dev.	Test	Classes
BIOMED	CHEMPROT	relation classification	4169	-	2427	3469	13
	†RCT	abstract sent. roles	18040	-	30212	30135	5
CS	ACL-ARC	citation intent	1688	-	114	139	6
	SCIERC	relation classification	3219	-	455	974	7
NEWS	HYPERPARTISAN	partisanship	515	5000	65	65	2
	†AGNEWS	topic	115000	-	5000	7600	4
REVIEWS	†HELPFULNESS	review helpfulness	115251	-	5000	25000	2
	†IMDB	review sentiment	20000	50000	5000	25000	2

Table 2: Specifications of the various target task datasets. † indicates high-resource settings. Sources: CHEMPROT (Kringelum et al., 2016), RCT (Dernoncourt and Lee, 2017), ACL-ARC (Jurgens et al., 2018), SCIERC (Luan et al., 2018), HYPERPARTISAN (Kiesel et al., 2019), AGNEWS (Zhang et al., 2015), HELPFULNESS (McAuley et al., 2015), IMDB (Maas et al., 2011).

Domain-Adaptive Pretraining (DAPT)

Domain-Adaptive Pretraining(DAPT) : continue pretraining RoBERTa on a large corpus of unlabeled domain-specific text

>> Analyzing Domain Similarity

Figure 2: Vocabulary overlap (%) between domains. PT denotes a sample from sources similar to RoBERTa's pretraining corpus. Vocabularies for each domain are created by considering the top 10K most frequent words (excluding stopwords) in documents sampled from each domain.

Domain-Adaptive Pretraining (DAPT)

>> Experiment

Dom.	Task	RoBa.	DAPT	¬DAPT
BM	ChemProt	$81.9_{1.0}$	$\mathbf{84.2}_{0.2}$	$79.4_{1.3}$
	†RCT	$87.2_{0.1}$	$\mathbf{87.6}_{0.1}$	$86.9_{0.1}$
CS	ACL-ARC	$63.0_{5.8}$	$\mathbf{75.4}_{2.5}$	$66.4_{4.1}$
	SciERC	$77.3_{1.9}$	$\mathbf{80.8}_{1.5}$	$79.2_{0.9}$
News	Hyp.	$86.6_{0.9}$	$\mathbf{88.2}_{5.9}$	$76.4_{4.9}$
	†AGNews	$\mathbf{93.9}_{0.2}$	$\mathbf{93.9}_{0.2}$	$93.5_{0.2}$
Rev.	†Helpful.	$65.1_{3.4}$	$\mathbf{66.5}_{1.4}$	$65.1_{2.8}$
	†IMDB	$95.0_{0.2}$	$\mathbf{95.4}_{0.2}$	$94.1_{0.4}$

Table 3: Comparison of RoBERTa (RoBa.) and DAPT to adaptation to an *irrelevant* domain (¬DAPT). Reported results are test macro-F_1, except for ChemProt and RCT, for which we report micro-F_1, following Beltagy et al. (2019). We report averages across five random seeds, with standard deviations as subscripts. † indicates high-resource settings. Best task performance is boldfaced. See §3.3 for our choice of irrelevant domains.

DAPT improves over RoBERTa in all domains

- domain relevance for DAPT (¬DAPT)
: DAPT significantly outperforms adapting to an irrelecant domain, suggesting the importance of pretraining on domain-relevant data

Task-Adaptive Pretraining (TAPT)

Task-Adaptive Pretraining(TAPT) : pretraining on the unlabeled training set for a given task

>>Experiment

Combined DAPT and TAPT

Domain	Task	RoBERTa	Additional Pretraining Phases		
			DAPT	TAPT	DAPT + TAPT
BioMed	ChemProt	$81.9_{1.0}$	$84.2_{0.2}$	$82.6_{0.4}$	$\mathbf{84.4}_{0.4}$
	†RCT	$87.2_{0.1}$	$87.6_{0.1}$	$87.7_{0.1}$	$\mathbf{87.8}_{0.1}$
CS	ACL-ARC	$63.0_{5.8}$	$75.4_{2.5}$	$67.4_{1.8}$	$\mathbf{75.6}_{3.8}$
	SciERC	$77.3_{1.9}$	$80.8_{1.5}$	$79.3_{1.5}$	$\mathbf{81.3}_{1.8}$
News	HyperPartisan	$86.6_{0.9}$	$88.2_{5.9}$	$\mathbf{90.4}_{5.2}$	$90.0_{6.6}$
	†AGNews	$93.9_{0.2}$	$93.9_{0.2}$	$94.5_{0.1}$	$\mathbf{94.6}_{0.1}$
Reviews	†Helpfulness	$65.1_{3.4}$	$66.5_{1.4}$	$68.5_{1.9}$	$\mathbf{68.7}_{1.8}$
	†IMDB	$95.0_{0.2}$	$95.4_{0.1}$	$95.5_{0.1}$	$\mathbf{95.6}_{0.1}$

Table 5: Results on different phases of adaptive pretraining compared to the baseline RoBERTa (col. 1). Our approaches are DAPT (col. 2, §3), TAPT (col. 3, §4), and a combination of both (col. 4). Reported results follow the same format as Table 3. State-of-the-art results we can compare to: ChemProt (84.6), RCT (92.9), ACL-ARC (71.0), SciERC (81.8), HyperPartisan (94.8), AGNews (95.5), IMDB (96.2); references in §A.2.

Cross-Task Tansfer

BioMed	RCT	ChemProt
TAPT	$87.7_{0.1}$	$82.6_{0.5}$
Transfer-TAPT	$87.1_{0.4}$ (↓0.6)	$80.4_{0.6}$ (↓2.2)
News	**HyperPartisan**	**AGNews**
TAPT	$89.9_{9.5}$	$94.5_{0.1}$
Transfer-TAPT	$82.2_{7.7}$ (↓7.7)	$93.9_{0.2}$ (↓0.6)

CS	ACL-ARC	SciERC
TAPT	$67.4_{1.8}$	$79.3_{1.5}$
Transfer-TAPT	$64.1_{2.7}$ (↓3.3)	$79.1_{2.5}$ (↓0.2)
Reviews	**Helpfulness**	**IMDB**
TAPT	$68.5_{1.9}$	$95.7_{0.1}$
Transfer-TAPT	$65.0_{2.6}$ (↓3.5)	$95.0_{0.1}$ (↓0.7)

Table 6: Though TAPT is effective (Table 5), it is harmful when applied *across* tasks. These findings illustrate differences in task distributions within a domain.

Augmenting Training Data for Task-Adaptive Pretraining

1 Human Curated-TAPT

Pretraining	BIOMED RCT-500	NEWS HYP.	REVIEWS IMDB †
TAPT	$79.8_{1.4}$	$90.4_{5.2}$	$95.5_{0.1}$
DAPT + TAPT	$83.0_{0.3}$	$90.0_{6.6}$	$95.6_{0.1}$
Curated-TAPT	$83.4_{0.3}$	$89.9_{9.5}$	$95.7_{0.1}$
DAPT + Curated-TAPT	$\mathbf{83.8}_{0.5}$	$\mathbf{92.1}_{3.6}$	$\mathbf{95.8}_{0.1}$

Table 7: Mean test set macro-F_1 (for HYP. and IMDB) and micro-F_1 (for RCT-500), with Curated-TAPT across five random seeds, with standard deviations as subscripts. † indicates high-resource settings.

2 Automated Data Selection for TAPT

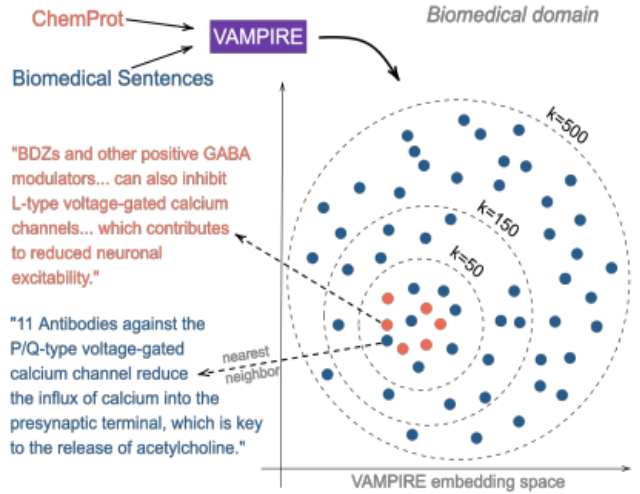


Figure 3: An illustration of automated data selection (§5.2). We map unlabeled CHEMPROT and 1M BIOMED sentences to a shared vector space using the VAMPIRE model trained on these sentences. Then, for each CHEMPROT sentence, we identify k nearest neighbors, from the BIOMED domain.

Pretraining	BIOMED CHEMPROT	BIOMED RCT-500	CS ACL-ARC
ROBERTA	$81.9_{1.0}$	$79.3_{0.6}$	$63.0_{5.8}$
TAPT	$82.6_{0.4}$	$79.8_{1.4}$	$67.4_{1.8}$
RAND-TAPT	$81.9_{0.6}$	$80.6_{0.4}$	$69.7_{3.4}$
50NN-TAPT	$83.3_{0.7}$	$80.8_{0.6}$	$70.7_{2.8}$
150NN-TAPT	$83.2_{0.6}$	$81.2_{0.8}$	$73.3_{2.7}$
500NN-TAPT	$83.3_{0.7}$	$81.7_{0.4}$	$\mathbf{75.5}_{1.9}$
DAPT	$\mathbf{84.2}_{0.2}$	$\mathbf{82.5}_{0.5}$	$75.4_{2.5}$

Table 8: Mean test set micro-F_1 (for CHEMPROT and RCT) and macro-F_1 (for ACL-ARC), across five random seeds, with standard deviations as subscripts, comparing RAND-TAPT (with 50 candidates) and kNN-TAPT selection. Neighbors of the task data are selected from the domain data.

Reference

Gururangan, Suchin, et al. "Don't stop pretraining: adapt language models to domains and tasks." *arXiv preprint arXiv:2004.10964* (2020).

Syn2Real Transfer Learning for Image Deraining

CVPR 2020

Rajeev Yasarla, Vishwanath A. Sindagi, Vishal M. Patel

Johns Hopkins University

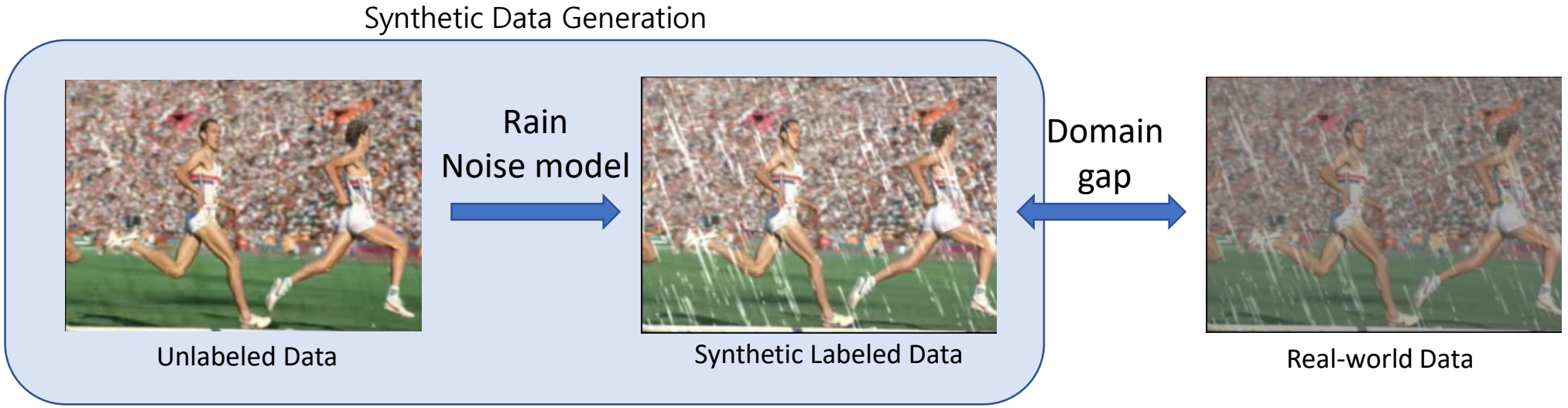
ISPL 박재현 2021-28335

Image Deraining

- Task of removing rain from single image

Problem: Lack of real-world labeled training data

Synthetic Generation of Labeled data



Synthetically generated input-output dataset pairs

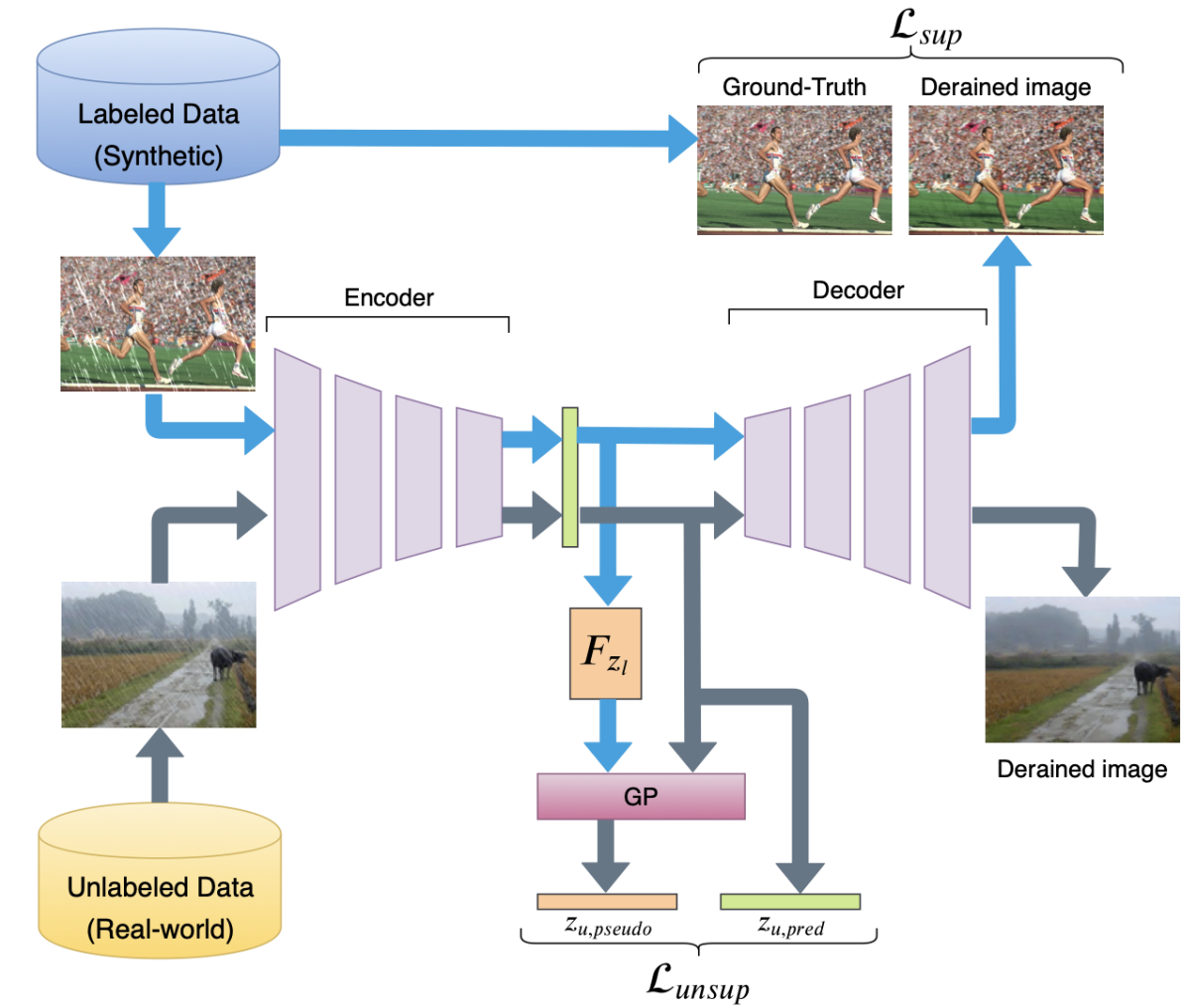
Problem: Suboptimal performance on real-world images due to domain gap

→ Need for Real-world images to improve **generalization** performance!

Paper’s Solution

- Semi-supervised Learning to incorporate unlabeled real-world data into training
- Gaussian Process-based Pseudo-labeling approach

- Iterative Multi-phase Training
  - Labeled Data training phase
  - Unlabeled Data training phase



Labeled path

Unlabeled path

F_{z_l} Labeled latent space vetcors

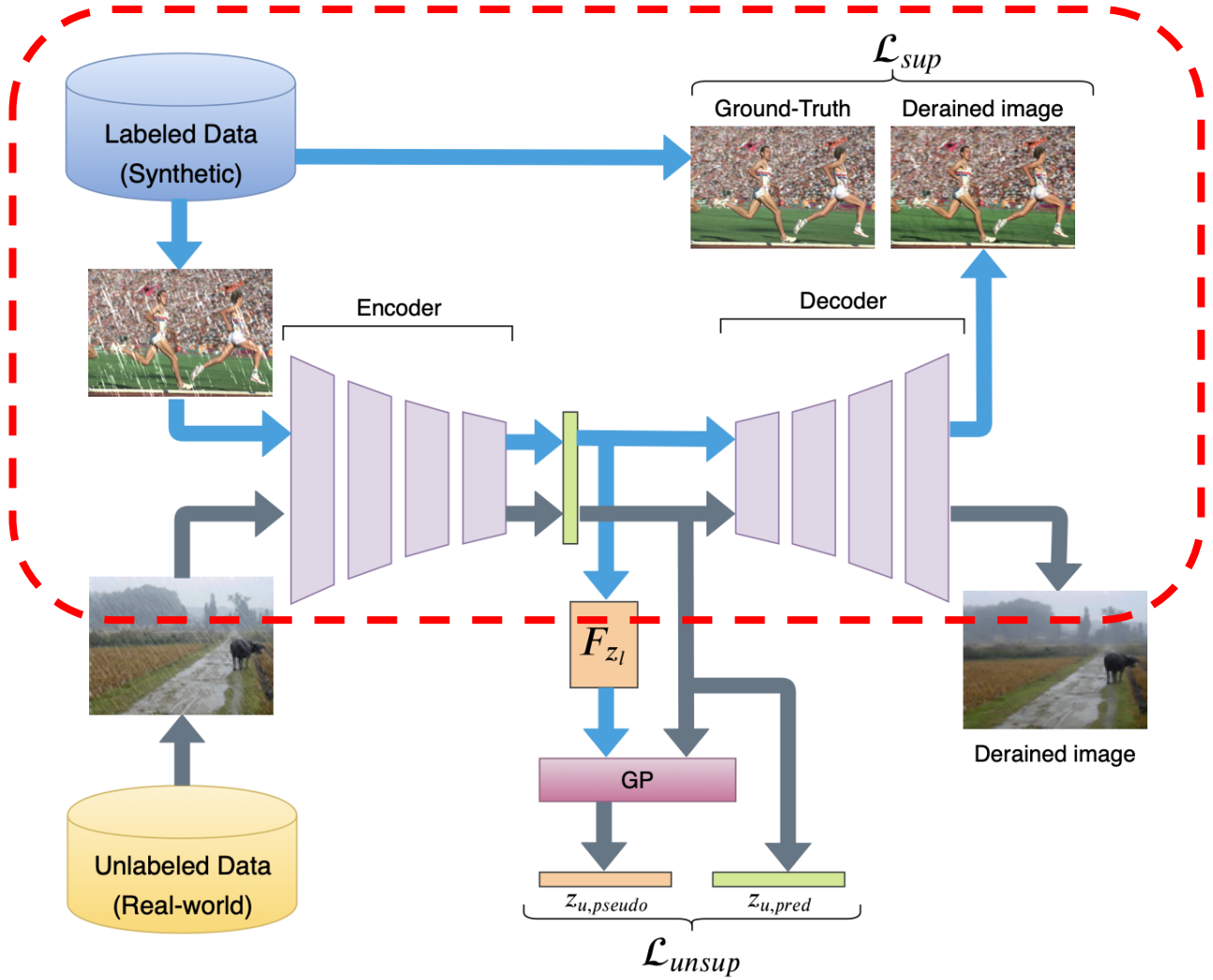
$z_{u,pseudo}$: pseudo-GT

$z_{u,pred}$ latent vector

$\mathcal{L}_{unsup}$ Loss for unlabeled data

$\mathcal{L}_{sup}$ = Loss for labeled data

GP = Gaussian Processes



Labeled path

Unlabeled path

F_{z_l} Labeled latent space vetcors

$z_{u,pseudo}$: pseudo-GT

$z_{u,pred}$ latent vector

$\mathcal{L}_{unsup}$ Loss for unlabeled data

$\mathcal{L}_{sup}$ = Loss for labeled data

GP = Gaussian Processes

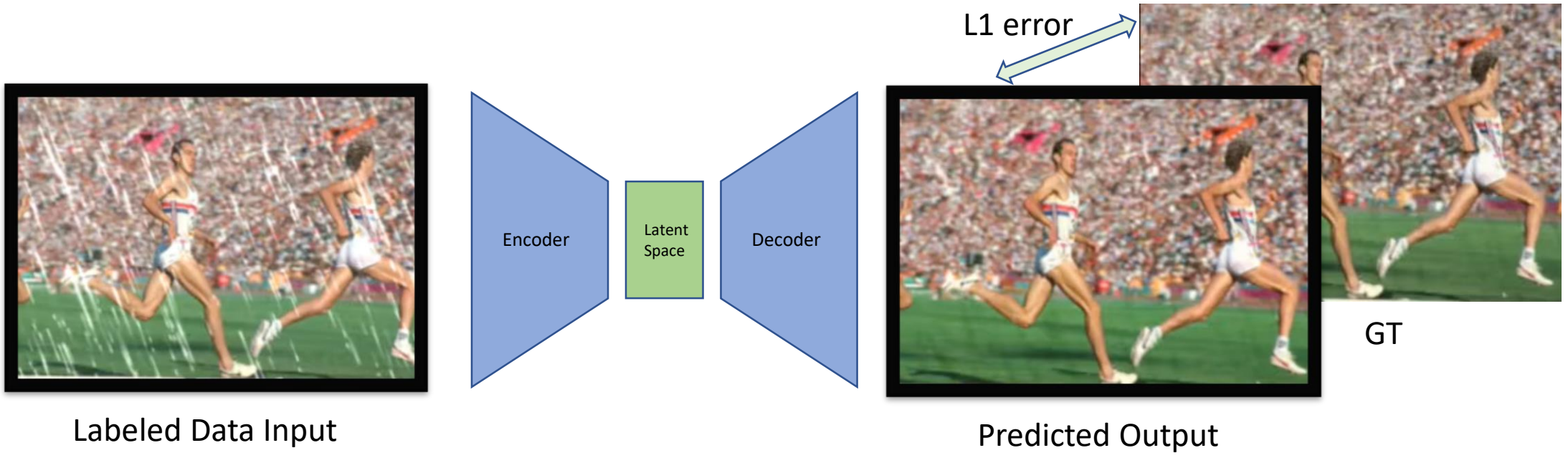
Labeled Data training phase

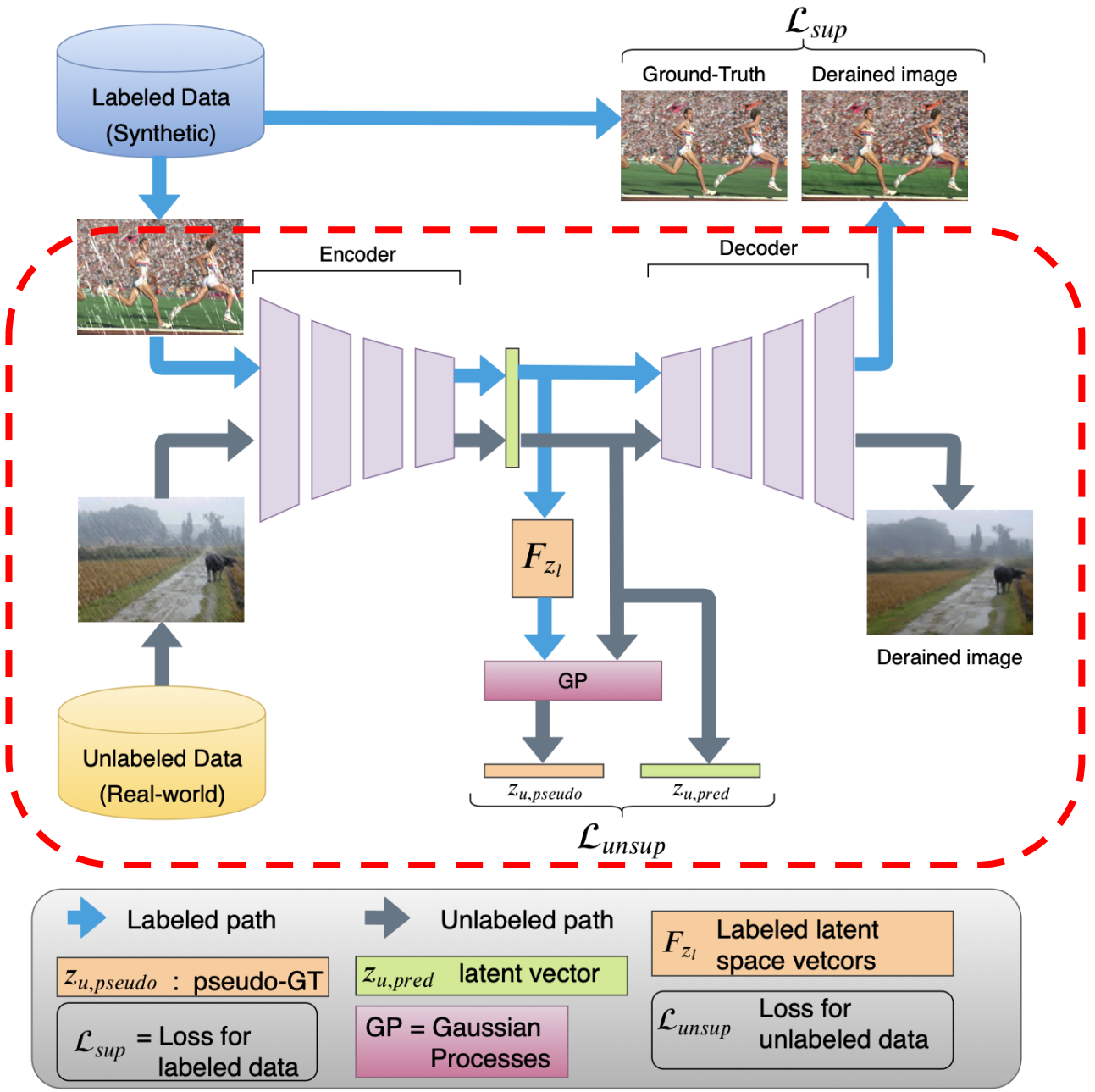
- Supervised Loss : Minimize error between predictions and GT

$$\mathcal{L}_{sup} = \mathcal{L}_1 + \lambda_p \mathcal{L}_p$$

$$\mathcal{L}_1 = \|y_l^{pred} - y_l\|_1,$$
$$\mathcal{L}_p = \|\Phi_{VGG}(y_l^{pred}) - \Phi_{VGG}(y_l)\|_2^2$$

- Inputs projected onto latent space are modeled using gaussian process and stored

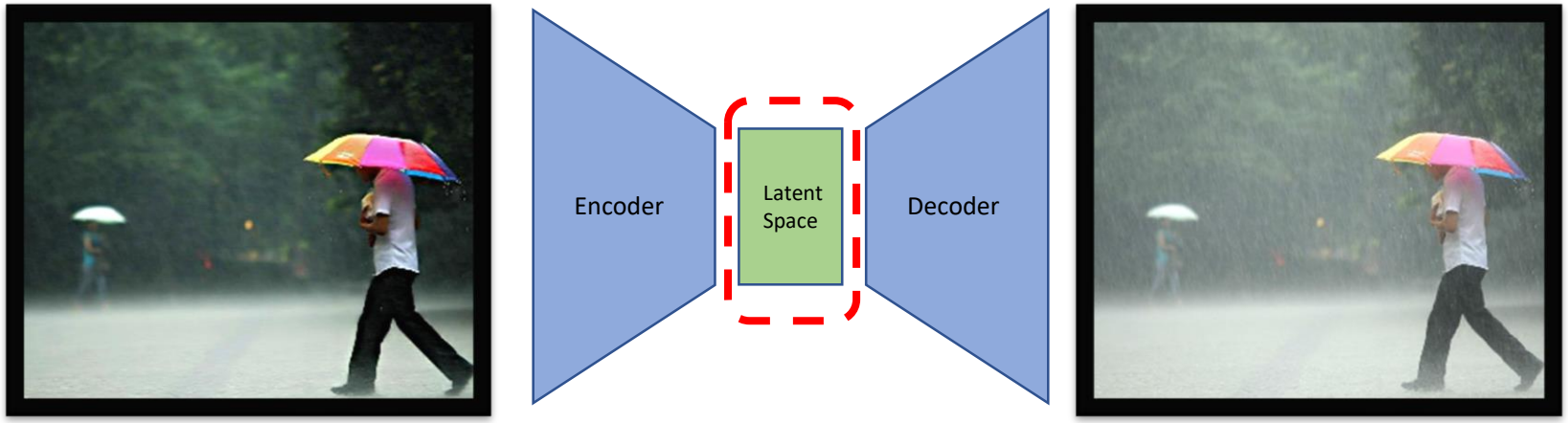



$\mathcal{L}_{sup}$
Ground-Truth
Derained image
Labeled Data
(Synthetic)
Encoder
Decoder
F_{z_l}
GP
Derained image
Unlabeled Data
(Real-world)
$z_{u,pseudo}$
$z_{u,pred}$
$\mathcal{L}_{unsup}$
Labeled path
Unlabeled path
F_{z_l} Labeled latent space vetcors
$z_{u,pseudo}$: pseudo-GT
$z_{u,pred}$ latent vector
$\mathcal{L}_{unsup}$ Loss for unlabeled data
$\mathcal{L}_{sup}$ = Loss for labeled data
GP = Gaussian Processes

Unlabeled Data training phase

- Unsupervised Loss : Minimize error at latent space between unlabeled data projections and pseudo-GT

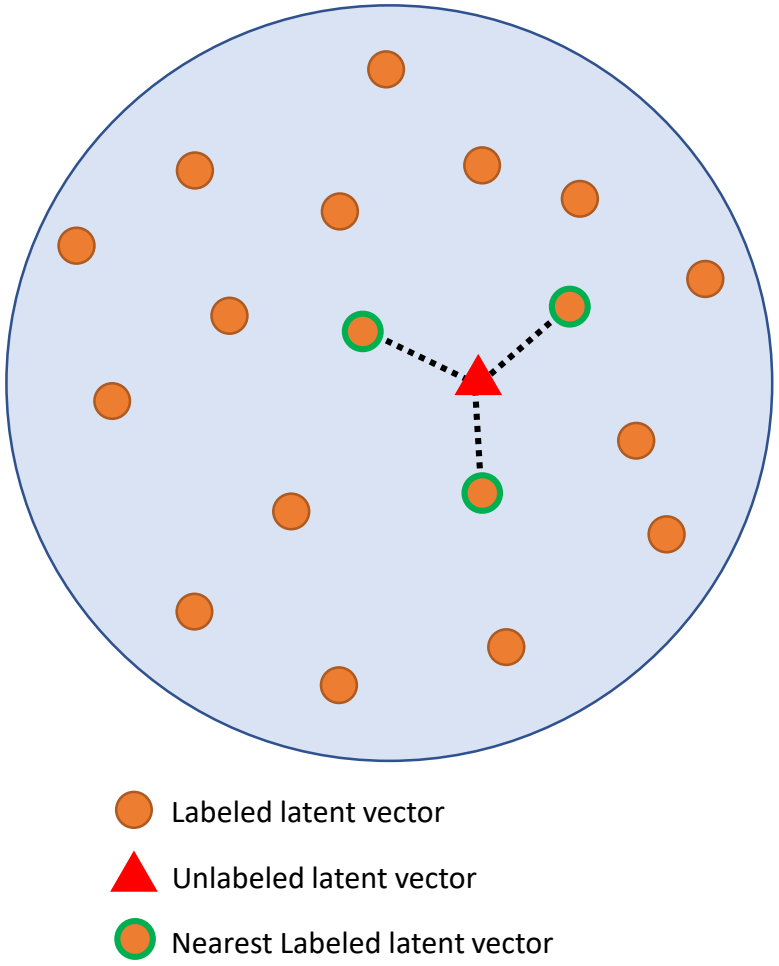
$$\mathcal{L}_{unsup} = \|z^k_{u,pred} - z^k_{u,pseudo}\|_2 + \log \Sigma^k_{u,n} + \log(1 - \Sigma^k_{u,f})$$



Unlabeled Data Input

Predicted Output

Latent Space Projections



- Encoder projects both labeled and unlabeled data into latent space
- Using kNN, unlabeled latent vectors can be represented as a linear combination of labeled latent vectors

$$z_u^k = \sum_{i=1}^{N_l} \alpha_i z_l^i + \epsilon,$$

Gaussian Process Modeling and Pseudo-GT

- Modeling joint distribution of nearest labeled latent vectors and unlabeled latent vector

$$P(z_u^k|\mathcal{D}_\mathcal{L}, F_{z_l}) = \mathcal{N}(\mu_u^k, \Sigma_u^k),$$

$$\mu_u^k = K(z_u^k, F_{z_l})[K(F_{z_l}, F_{z_l}) + \sigma_\epsilon^2 I]^{-1} F_{z_l},$$

→ Pseudo-GT

$$\Sigma_u^k = K(z_u^k, z_u^k) - K(z_u^k, F_{z_l})[K(F_{z_l}, F_{z_l}) + \sigma_\epsilon^2 I]^{-1}$$
$$K(F_{z_l}, z_u^k) + \sigma_\epsilon^2$$

- Minimize error between Pseudo-GT and unlabeled latent vector, update the weights of the encoder
- Adapts network to unlabeled data, hence results in better generalization

$$\mathcal{L}_{unsup} = \|z_{u,pred}^k - z_{u,pseudo}^k\|_2 + \log \Sigma_{u,n}^k + \log(1 - \Sigma_{u,f}^k)$$

Experiment Results

- Results on Synthetic test set

Table 1. Effect of using unlabeled real-world data in training process on DDN-SIRR dataset. Evaluation is performed on synthetic dataset similar to [49]. Proposed method achieves better gain in PSNR as compared to SIRR[49] in the case of both Dense and Sparse categories. $\mathcal{D}_{\mathcal{L}}$ indicates training using only labeled dataset and $\mathcal{D}_{\mathcal{L}} + \mathcal{D}_{\mathcal{U}}$ indicates training using both labeled and unlabeled dataset.

Dataset	Input	Methods that use only synthetic dataset							Methods that use synthetic and real-world dataset					
		DSC [30] (ICCV '15)	LP [24] (CVPR '16)	JORDER [51] (CVPR '17)	DDN [9] (CVPR '17)	JBO [60] (CVPR '17)	DID-MDN [58] (CVPR '18)	UMRL [53] (CVPR '19)	SIRR [49] (CVPR '19)			Ours		
									$\mathcal{D}_{\mathcal{L}}$	$\mathcal{D}_{\mathcal{L}} + \mathcal{D}_{\mathcal{U}}$	Gain	$\mathcal{D}_{\mathcal{L}}$	$\mathcal{D}_{\mathcal{L}} + \mathcal{D}_{\mathcal{U}}$	Gain
Dense	17.95	19.00	19.27	18.75	19.90	18.87	18.60	20.11	20.01	21.60	1.59	20.24	**22.36**	**2.12**
Sparse	24.14	25.05	25.67	24.22	26.88	25.24	25.66	26.94	26.90	26.98	0.08	26.15	**27.26**	**1.11**

(a) (b) (c) (d) (e) (f)

Figure 4. Qualitative results on DDN-SIRR **synthetic** test set. (a) Input rainy image (b) DID-MDN [58](CVPR '18) (c) DDN [9](CVPR '17) (d) SIRR [49](CVPR '19) (e) Ours (f) ground-truth image.

Experiment Results

- Results on Real world test set

Metrics	Input	SIRR [49]			Ours		
		$\mathcal{D}_{\mathcal{L}}$	$\mathcal{D}_{\mathcal{L}} + \mathcal{D}_{\mathcal{U}}$	Gain	$\mathcal{D}_{\mathcal{L}}$	$\mathcal{D}_{\mathcal{L}} + \mathcal{D}_{\mathcal{U}}$	Gain
NIQE	4.671	3.86	3.84	0.02	3.85	3.78	0.07
BRISQUE	31.37	26.61	25.29	1.32	25.77	22.95	2.82

(a) (b) (c) (d) (e)

Figure 5. Qualitative results on DDN-SIRR **real-world** test set. (a) Input rainy image (b) DID-MDN [58] (c) DDN [9] (d) SIRR [49] (e) Ours.

GraphCodeBERT : Pre-Training Code Representations with Data Flow

HCC Lab
박희강
2021-22918

Introduction
: Paper

- GraphCodeBERT : Pre-Training Code Representations with Data Flow
  - Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Alexey Svyatkovskiy, Shengyu Fu, Michele Tufano, Shao Kun Deng, Colin Clement, Dawn Drain, Neel Sundaresan, Jian Yin, Daxin Jiang, Ming Zhou
  - ICLR 2021
  - Paper : https://arxiv.org/pdf/2009.08366.pdf
  - Code & Dataset : https://github.com/microsoft/CodeBERT/tree/master/GraphCodeBERT

Introduction
: Domain

- Code Representation
  - Source Code에 대한 Language Model을 만드는 분야
  - Compiler 이론, NLP, HCI 등 다양한 분야와 연관

- 이렇게 만들어진 Code Representation을 이용하면 다양한 Task를 할 수 있다.
  - Code Search : 자연어로 된 Query가 주어졌을 때, 주어진 후보 Code Snippet 안에서 이와 가장 유사한(semantically related) Code Snippet을 반환하는 Task
  - Clone Detection : 두 Code Snippet의 유사도(similarity)를 측정하는 Task
  - Code Translation : 한 Programming Language로 작성되어 있는 Code Snippet을 다른 Programming Language로 변환하는 Task
  - Code Refinement : Code Snippet에 있는 버그를 자동으로 탐지/수정하는 Task

Previous Works

- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
  - Paper : https://arxiv.org/pdf/1810.04805.pdf
  - NAACL 2019
  - “BERT” 류 학습법을 제시

- CodeBERT: A Pre-Trained Model for Programming and Natural Languages
  - Paper : https://arxiv.org/pdf/2002.08155.pdf
  - EMNLP 2020
  - Code Representation 분야에 BERT를 최초로 적용

- CodeSearchNet Challenge: Evaluating the State of Semantic Code Search
  - Paper : https://arxiv.org/pdf/1909.09436.pdf
  - 2019
  - Code Search Task를 위한 학습 데이터셋 “CodeSearchNet Corpus”와 평가 메트릭 “CodeSearchNet Challenge”를 제안
  - GraphCodeBERT도 CodeSearchNet Corpus를 이용해 학습되었음

Key Contribution

- CodeBERT + Data Flow
  - CodeBERT에서는 Source Code의 Semantic-level Structure에 대한 고려 없이, Source Code를 그저 Token들의 Sequence로 대함
    - Ex) `v = max_value - min_value`
    - CodeBERT에서는, "max", "value", "min", "value" (sub)token들로부터 이 Code가 최대값과 최소값의 차를 구하는 Code임을 이해함
    - 만약 프로그래머가 변수명으로 “max_value”, “min_value”와 같은 이름을 사용하지 않았다면, CodeBERT는 이 Code가 어떤 의미(semantic)를 가지는지 이해하지 못함
  - GraphCodeBERT는 Semantic-level Structure에 대한 정보를 담고 있는 Data Flow를 학습 과정에서 참조하게 해 이 문제를 해결

Result

model	Ruby	Javascript	Go	Python	Java	Php	Overall
NBow	0.162	0.157	0.330	0.161	0.171	0.152	0.189
CNN	0.276	0.224	0.680	0.242	0.263	0.260	0.324
BiRNN	0.213	0.193	0.688	0.290	0.304	0.338	0.338
selfAtt	0.275	0.287	0.723	0.398	0.404	0.426	0.419
RoBERTa	0.587	0.517	0.850	0.587	0.599	0.560	0.617
RoBERTa (code)	0.628	0.562	0.859	0.610	0.620	0.579	0.643
CodeBERT	0.679	0.620	0.882	0.672	0.676	0.628	0.693
GraphCodeBERT	**0.703**	**0.644**	**0.897**	**0.692**	**0.691**	**0.649**	**0.713**

Table 1: Results on code search. GraphCodeBERT outperforms other models significantly ($p < 0.01$).

Model	Precision	Recall	F1
Deckard	0.93	0.02	0.03
RtvNN	0.95	0.01	0.01
CDLH	0.92	0.74	0.82
ASTNN	0.92	0.94	0.93
FA-AST-GMN	**0.96**	0.94	0.95
RoBERTa (code)	0.949	0.922	0.935
CodeBERT	0.947	0.934	0.941
GraphCodeBERT	0.948	**0.952**	**0.950**

Table 2: Results on code clone detection. GraphCodeBERT outperforms other pre-trained methods significantly ($p < 0.01$).

Method	Java→C#		C#→Java	
	BLEU	Acc	BLEU	Acc
Naive	18.54	0.0	18.69	0.0
PBSMT	43.53	12.5	40.06	16.1
Transformer	55.84	33.0	50.47	37.9
RoBERTa (code)	77.46	56.1	71.99	57.9
CodeBERT	79.92	59.0	72.14	58.8
GraphCodeBERT	**80.58**	**59.4**	**72.64**	**58.8**

Table 3: Results on code translation.

Method	small		medium	
	BLEU	Acc	BLEU	Acc
Naive	78.06	0.0	90.91	0.0
LSTM	76.76	10.0	72.08	2.5
Transformer	77.21	14.7	89.25	3.7
RoBERTa (code)	77.30	15.9	90.07	4.1
CodeBERT	77.42	16.4	91.07	5.2
GraphCodeBERT	**80.02**	**17.3**	**91.31**	**9.1**

Table 4: Results on code refinement.

Thank you

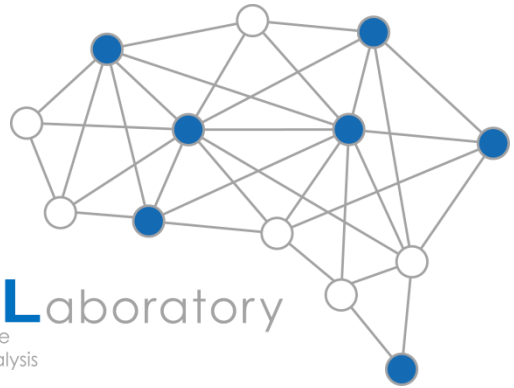
감사합니다.

Human Pose Regression with Residual Log-likelihood Estimation

Jiefeng Li, Siyuan Bian, Ailing Zeng, Can Wang, Bo Pang, Wentao Liu, Cewu Lu
Shanghai Jiao Tong University, The Chinese University of Hong Kong, SenseTime Research

ICCV 2021 Oral

발표자: 서승현



MIPAL aboratory
Machine Intelligence
& Pattern Analysis

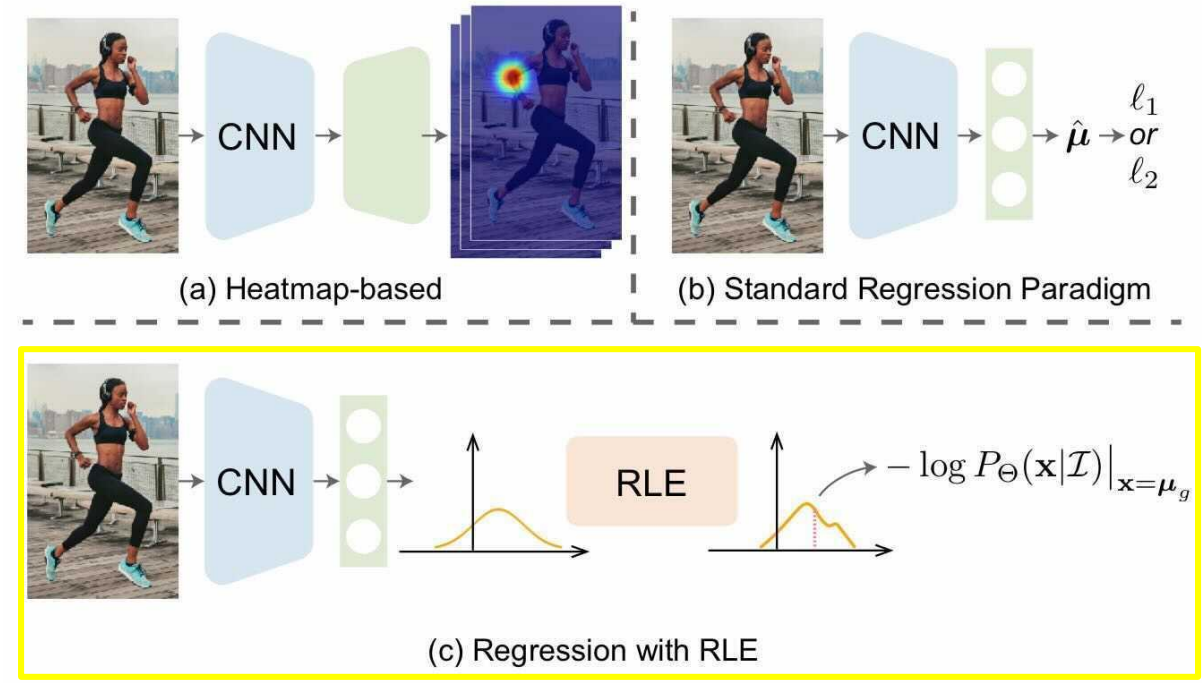
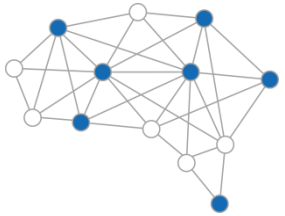


Figure 1: **Illustrasion** of (a) heatmap-based method, (b) standard regression paradigm, and (c) regression with the proposed RLE.

General Formulation of Regression



$$\mathcal{L}_{mle} = -\log P_{\Theta}(\mathbf{x}|\mathcal{I})\Big|_{\mathbf{x}=\boldsymbol{\mu}_g}$$

probability of gt appearing in the location X

$$P_{\Theta}(\mathbf{x}|\mathcal{I}) = \frac{1}{\sqrt{2\pi}\hat{\boldsymbol{\sigma}}} e^{-\frac{(\mathbf{x}-\hat{\boldsymbol{\mu}})^2}{2\hat{\boldsymbol{\sigma}}^2}}$$

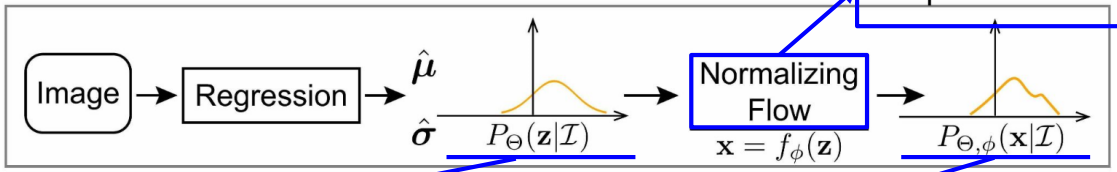
model's output

$$\mathcal{L} = -\log P_{\Theta}(\mathbf{x}|\mathcal{I})\Big|_{\mathbf{x}=\boldsymbol{\mu}_g} \propto \log \hat{\boldsymbol{\sigma}} + \frac{(\boldsymbol{\mu}_g - \hat{\boldsymbol{\mu}})^2}{2\hat{\boldsymbol{\sigma}}^2}$$

gaussian with constant variance: L2 loss
laplace with constant variance: L1 loss

Regression with Normalizing Flows

Basic Design



simple distribution → invertible mapping → complex distribution

$$P_\Theta(\mathbf{z}|\mathcal{I}) = \frac{1}{\sqrt{2\pi}\hat{\boldsymbol{\sigma}}} e^{-\frac{(\mathbf{z}-\hat{\boldsymbol{\mu}})^2}{2\hat{\boldsymbol{\sigma}}^2}}$$

Θ: regression model
f_ϕ: flow model

$$\log P_{\Theta,\phi}(\mathbf{x}|\mathcal{I}) = \log P_\Theta(\mathbf{z}|\mathcal{I}) + \log \left| \det \frac{\partial f_\phi^{-1}}{\partial \mathbf{x}} \right|$$

f_ϕ^{-1} is the inverse of f_ϕ and $\mathbf{z} = f_\phi^{-1}(\mathbf{x})$.

$$\mathcal{L}_{mle} = -\log P_{\Theta,\phi}(\mathbf{x}|\mathcal{I})\Big|_{\mathbf{x}=\boldsymbol{\mu}_g}$$

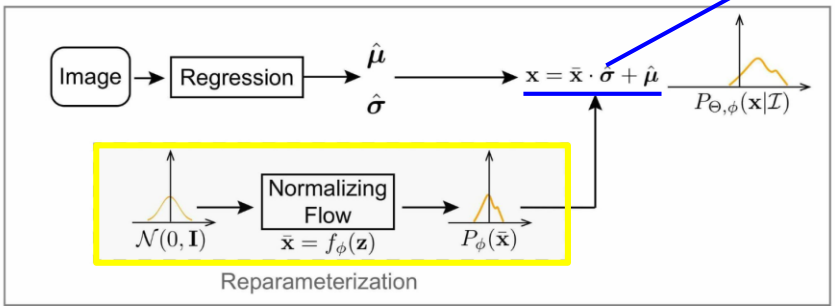
$$= -\log P_\Theta(f_\phi^{-1}(\boldsymbol{\mu}_g) \mid \mathcal{I}) - \log \left| \det \frac{\partial f_\phi^{-1}}{\partial \boldsymbol{\mu}_g} \right|$$

ϕ will learn to fit the distribution of $\boldsymbol{\mu}_g$ across all images

Regression with Normalizing Flows

Reparameterization

shifting / rescaling



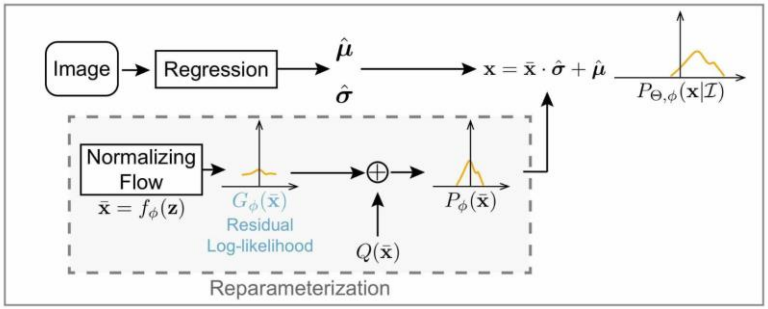
$$\mathcal{L}_{mle} = -\log P_{\Theta,\phi}(\mathbf{x}|\mathcal{I})\Big|_{\mathbf{x}=\boldsymbol{\mu}_g}$$
$$= -\log P_\phi(\bar{\boldsymbol{\mu}}_g) - \log\left|\det \frac{\partial \bar{\boldsymbol{\mu}}_g}{\partial \boldsymbol{\mu}_g}\right|$$
$$= -\log P_\phi(\bar{\boldsymbol{\mu}}_g) + \log \hat{\boldsymbol{\sigma}}, \qquad \text{where } \bar{\boldsymbol{\mu}}_g = (\boldsymbol{\mu}_g - \hat{\boldsymbol{\mu}})/\hat{\boldsymbol{\sigma}}, \text{ and } \partial\bar{\boldsymbol{\mu}}_g/\partial\boldsymbol{\mu}_g = 1/\hat{\boldsymbol{\sigma}}.$$

flow model f_ϕ can focus on learning the distribution of μ_bar_g

training of the regression model entirely relies on the distribution estimated by the flow model f_ϕ

Regression with Normalizing Flows

Residual Log-likelihood Estimation



optimal underlying distribution

simple distribution e.g. gaussian

$$\log P_{opt}(\bar{\mathbf{x}}) = \log\left(Q(\bar{\mathbf{x}}) \cdot \frac{P_{opt}(\bar{\mathbf{x}})}{s \cdot Q(\bar{\mathbf{x}})} \cdot s\right)$$
$$= \log Q(\bar{\mathbf{x}}) + \log \frac{P_{opt}(\bar{\mathbf{x}})}{s \cdot Q(\bar{\mathbf{x}})} + \log s,$$

residual log-likelihood

$$\log P_\phi(\bar{\mathbf{x}}) = \log Q(\bar{\mathbf{x}}) + \log G_\phi(\bar{\mathbf{x}}) + \log s,$$

distribution learned by the flow model.
It will try to fit the underlying residual likelihood instead of learning the entire distribution.

$$\mathcal{L}_{rle} = -\log P_{\Theta,\phi}(\mathbf{x}|\mathcal{I})\Big|_{\mathbf{x}=\boldsymbol{\mu}_g}$$
$$= -\log P_\phi(\bar{\boldsymbol{\mu}}_g) + \log \hat{\boldsymbol{\sigma}}$$
$$= -\log Q(\bar{\boldsymbol{\mu}}_g) - \log G_\phi(\bar{\boldsymbol{\mu}}_g) - \log s + \log \hat{\boldsymbol{\sigma}}.$$

As the hypothesis of ResNet, it is easier to optimize the residual mapping than to optimize the original unreferenced mapping.

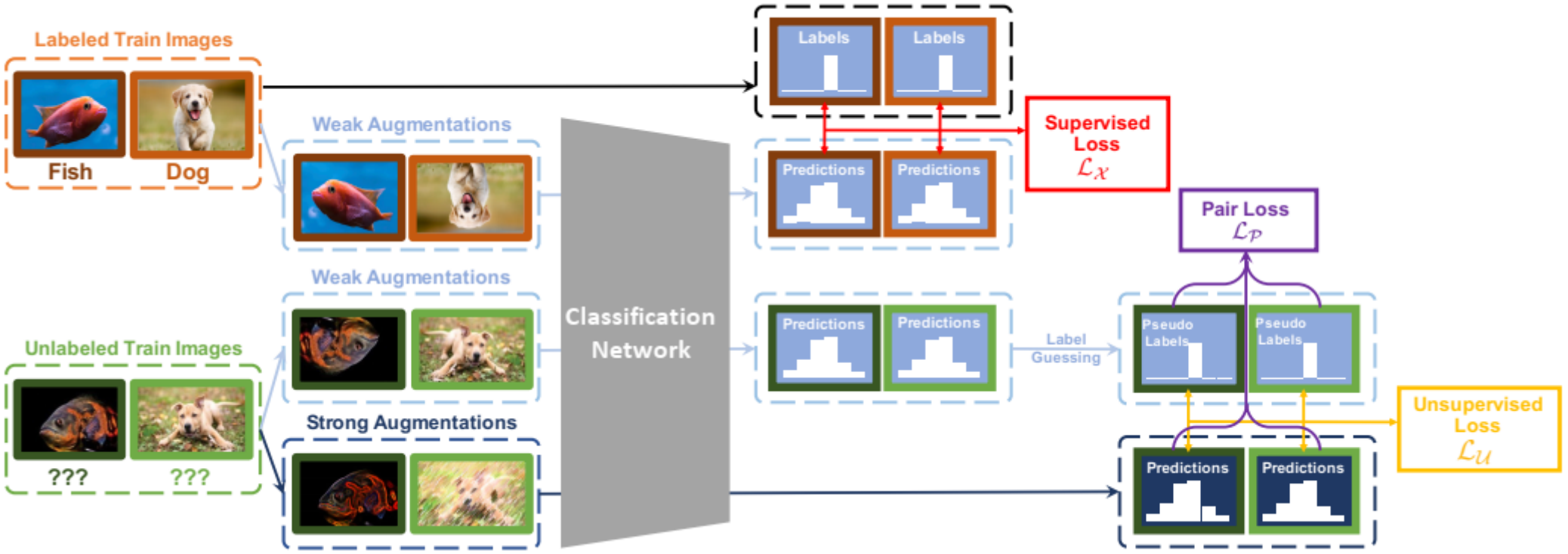
SimPLE: Similar Pseudo Label Exploitation for Semi-Supervised Classification

202134942 송우혁

Introduction

- Semi-supervised training
- SimPLE algorithm: Use information from unlabeled data
  - Proposed “pair loss”

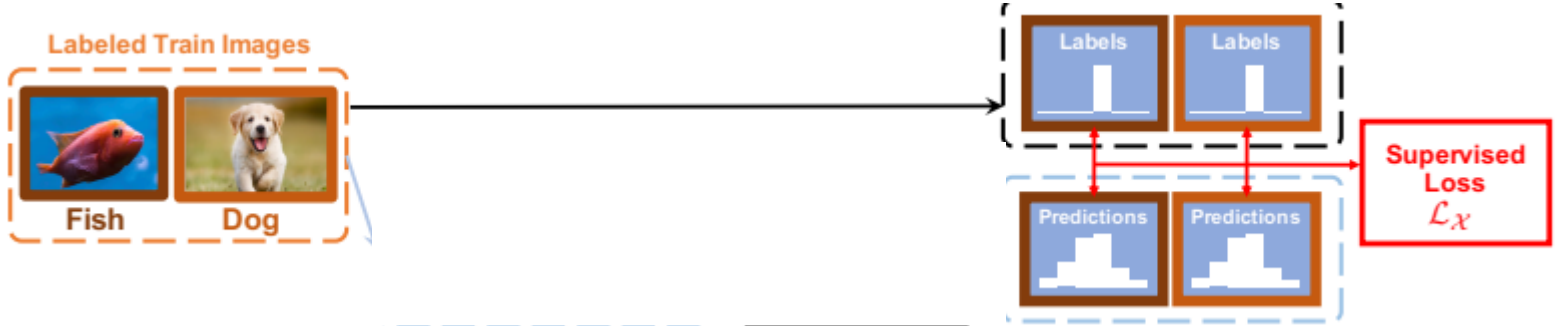
Overview


Labeled Train Images
Fish
Dog
Unlabeled Train Images
???
???
Weak Augmentations
Weak Augmentations
Strong Augmentations
Classification Network
Labels
Labels
Predictions
Predictions
Predictions
Predictions
Supervised Loss
$\mathcal{L}_{\mathcal{X}}$
Label Guessing
Pair Loss
$\mathcal{L}_{\mathcal{P}}$
Pseudo Labels
Pseudo Labels
Predictions
Predictions
Unsupervised Loss
$\mathcal{L}_{\mathcal{U}}$

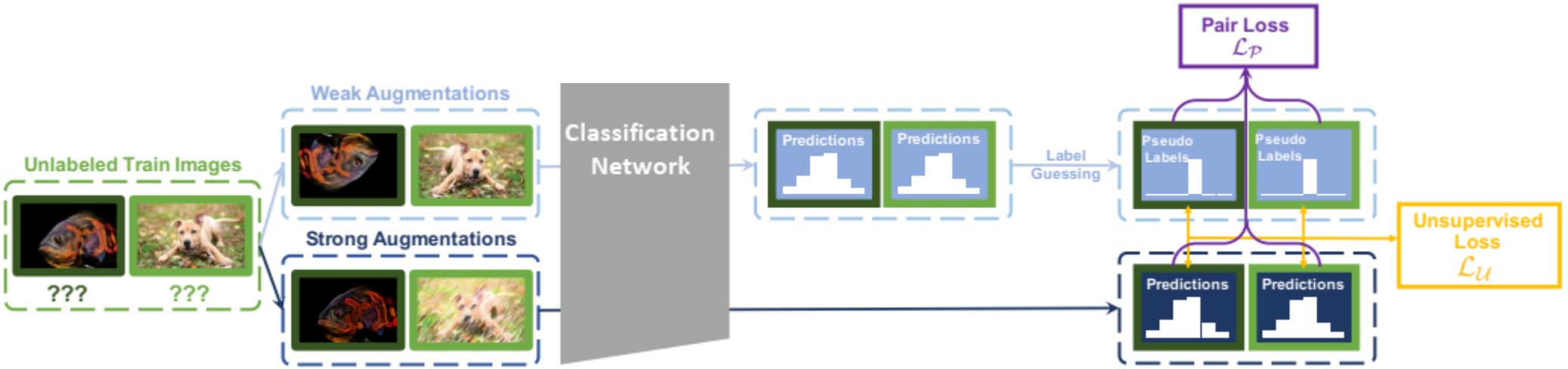
Method

- Augmentation strategy
  - Anchor: pseudo labels from weakly augmented samples
- Pseudo-labeling
  - Pseudo label: Model prediction average of several weakly augmented versions of same unlabeled sample

Method

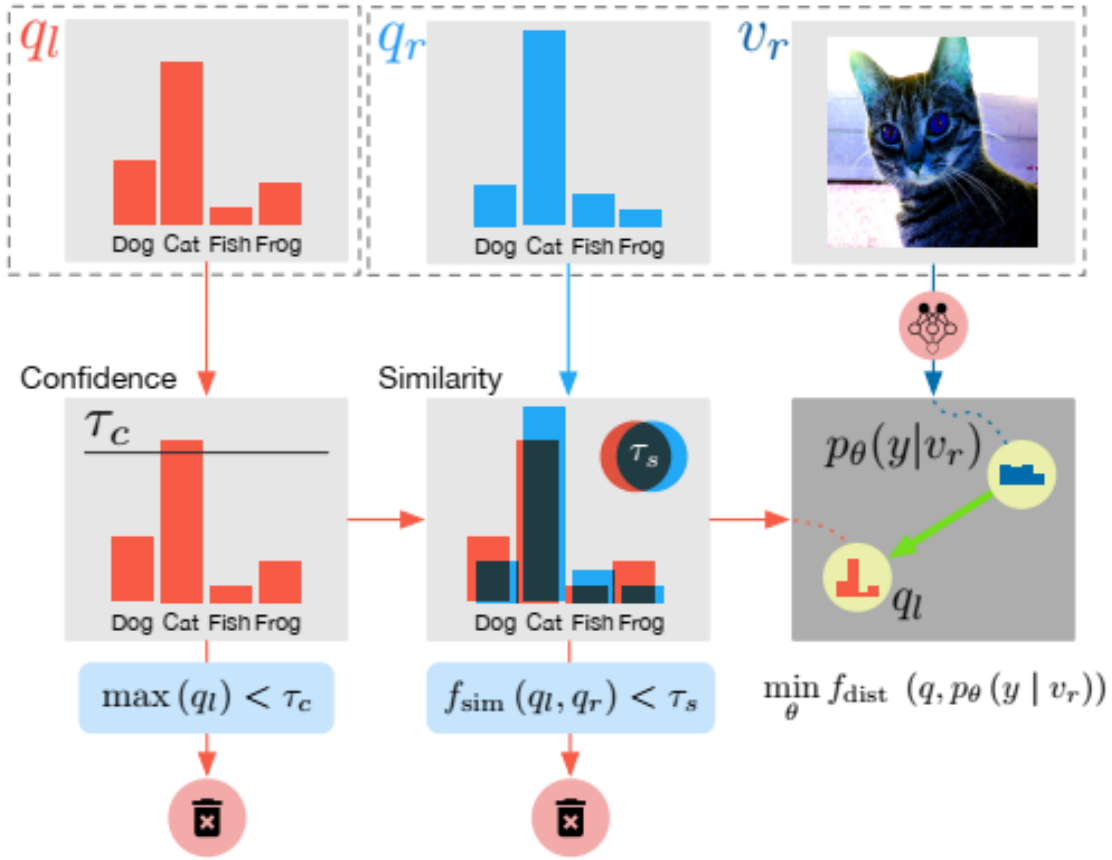


- Loss
  - Supervised loss: Cross-entropy of weakly augmented labeled samples
  - Unsupervised loss: L2 distance btw strongly augmented samples and their pseudo labels



Method

- Loss
  - Pair loss



Experiments

- Transfer learning
  - Performs well from pre-trained model on different dataset
  - Fast convergence

Transfer: DomainNet-Real to Mini-ImageNet		
Method	4000 labels	Convergence step
Supervised w/ EMA[§]	48.83%	4K
MixMatch* from scratch	50.31%	150K
MixMatch*	53.39%	69K
MixMatch Enhanced* from scratch	52.83%	734K
MixMatch Enhanced*	55.75%	7K
SimPLE from scratch	**59.92%**	338K
SimPLE	**58.73%**	53K

Table 4: DomainNet-Real pre-trained model transfer to Mini-ImageNet. All experiments use WRN 28-2. The model is converged when its validation accuracy reaches 95% of its highest validation accuracy. [§]: using labeled training set only. *: using our implementation.

Course: Pattern Recognition

FlexMatch: Boosting Semi-Supervised Learning with Curriculum Pseudo Labeling

NeurIPS 2021

신주현 (Juhyeon Shin)

협동과정 인공지능 전공

Seoul National University

Data Science & Artificial Intelligence Laboratory

2021.11.19

DSAIL
Data Science & AI Laboratory

Introduction

FlexMatch: Boosting Semi-Supervised Learning with Curriculum Pseudo Labeling

Bowen Zhang[*]
Tokyo Institute of Technology
bowen.z.ab@m.titech.ac.jp

Yidong Wang[*]
Tokyo Institute of Technology
wang.y.ca@m.titech.ac.jp

Wenxin Hou
Microsoft
wenxinhou@microsoft.com

Hao Wu
Tokyo Institute of Technology
wu.h.aj@m.titech.ac.jp

Jindong Wang[†]
Microsoft Research Asia
jindwang@microsoft.com

Manabu Okumura[†]
Tokyo Institute of Technology
oku@pi.titech.ac.jp

Takahiro Shinozaki[†]
Tokyo Institute of Technology
shinot@ict.e.titech.ac.jp

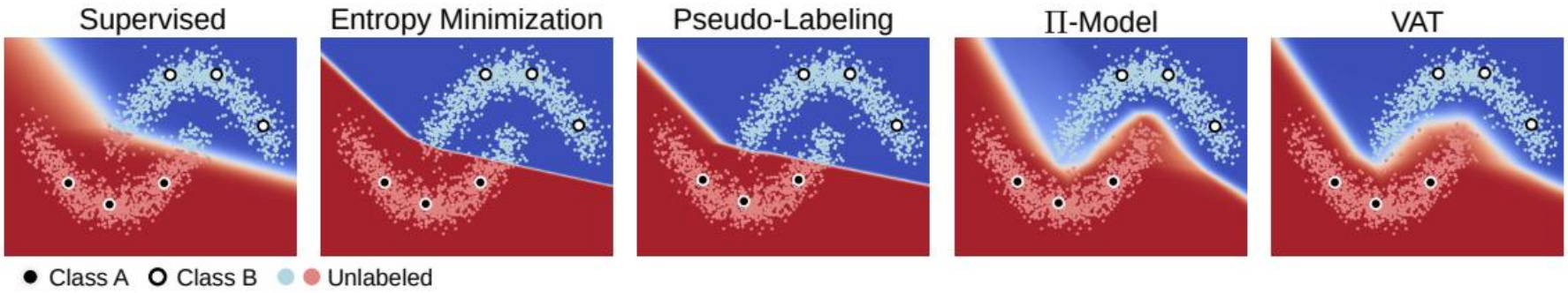
- Abstract
  - Propose **Curriculum Pseudo Labeling (CPL)**, a curriculum learning approach of dynamically leveraging unlabeled data for SSL
  - CPL is cost free and also significantly boosts the convergence speed
  - **FlexMatch (FixMatch + CPL)** achieves SOTA performance on a variety of SSL benchmarks

Backgrounds

- **Semi-Supervised learning (SSL)**: Learning with small amount of labeled and large amount of unlabeled data (1-stage)

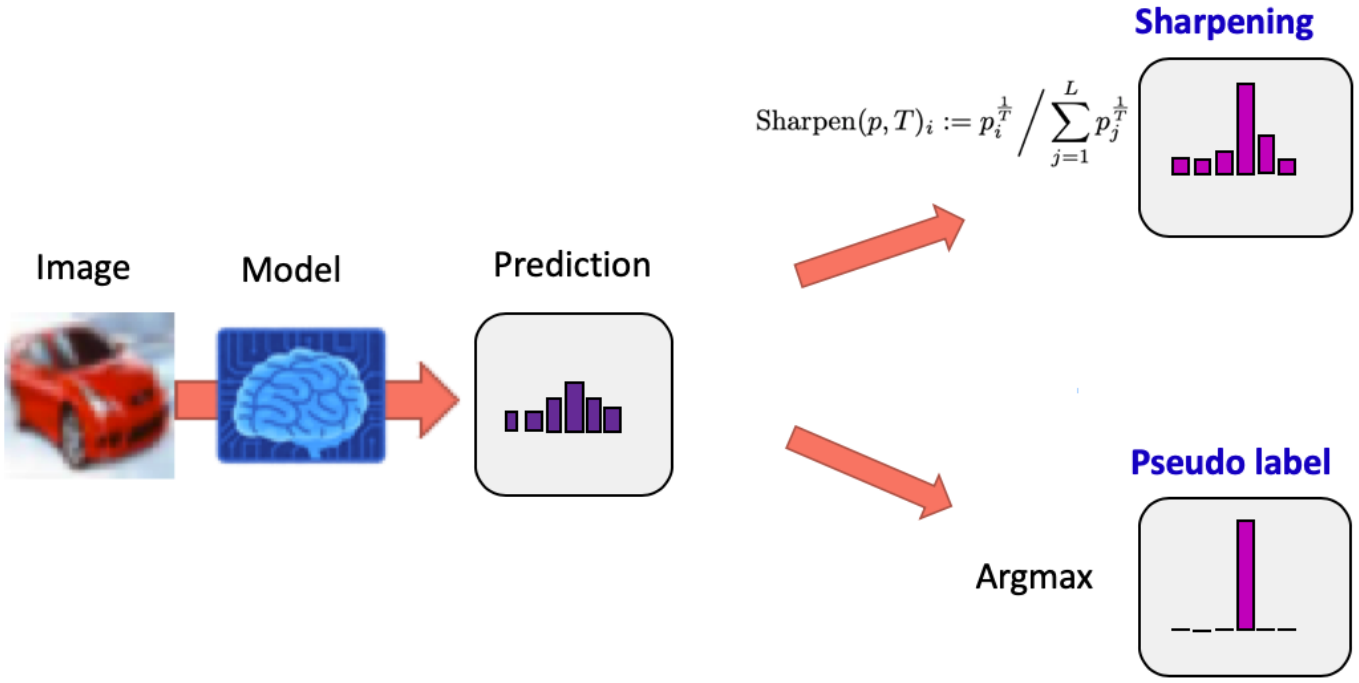
$$Loss = L_s + L_u$$

- Assumption
  - The smoothness assumption: points that are close to each other are more likely to share a label
  - The cluster assumption: If points are in the same cluster, they are likely to be of the same class
  - The manifold assumption: The (high-dim) data lie (roughly) on a low-dim manifold



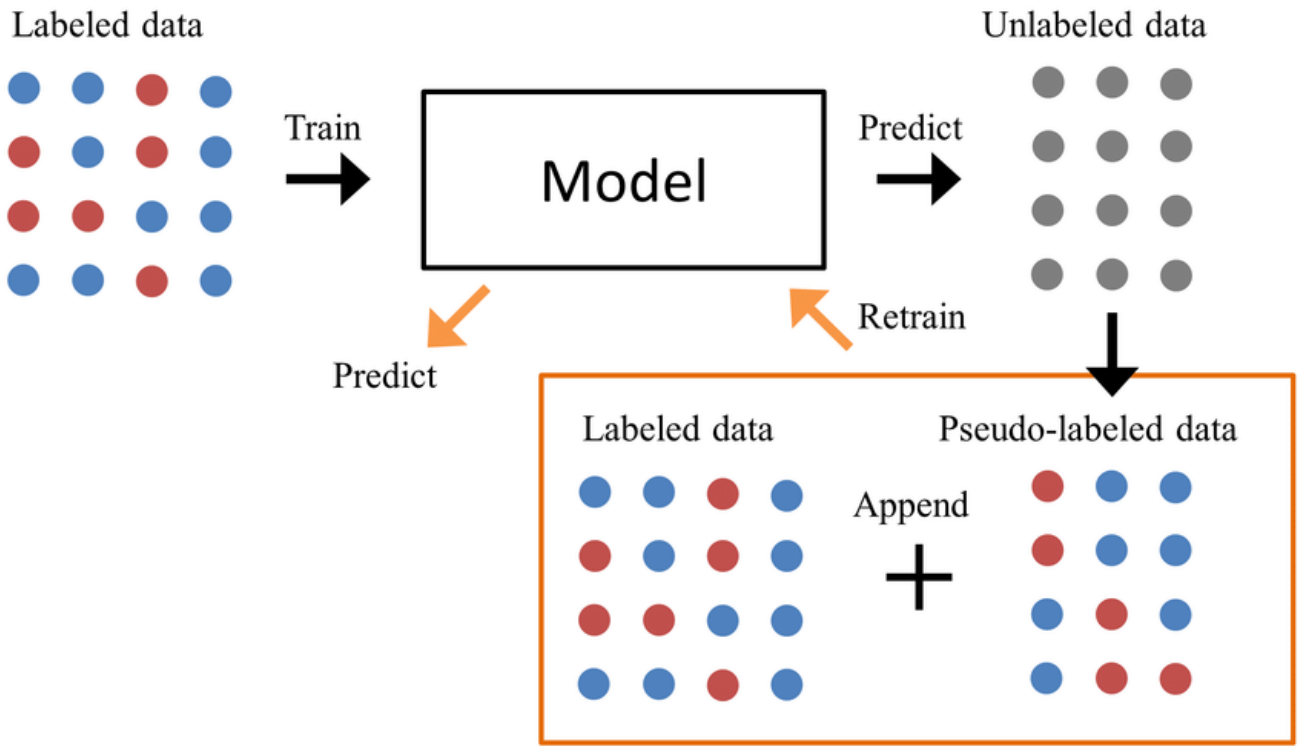
Backgrounds

- **Entropy minimization**
  - Decision boundary should not pass high-density regions of the marginal data distribution



Backgrounds

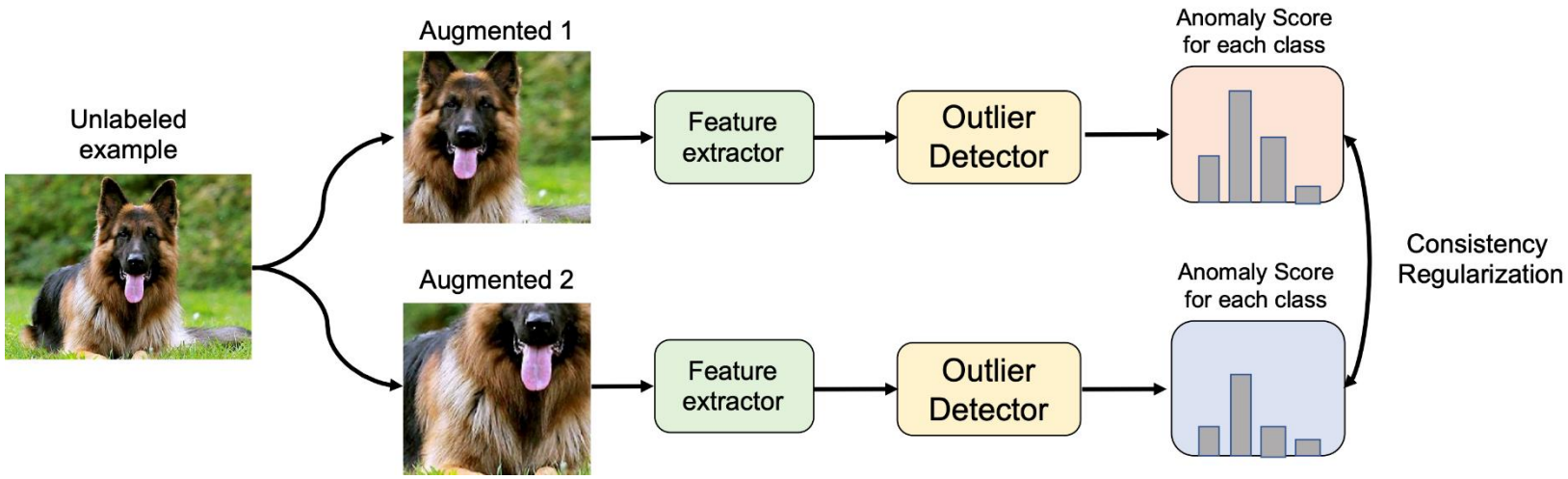
- **Pseudo Labeling** (self-training)
  - Entropy minimization approach
  - Use the model's class prediction as a label to train against



Backgrounds

- **Consistency regularization**
  - Enforce network predictions to be consistent when its input is perturbed

$$\sum_{b=1}^{\mu B} ||p_m(y|\omega(u_b)) - p_m(y|\omega(u_b))||_2^2$$



Related work

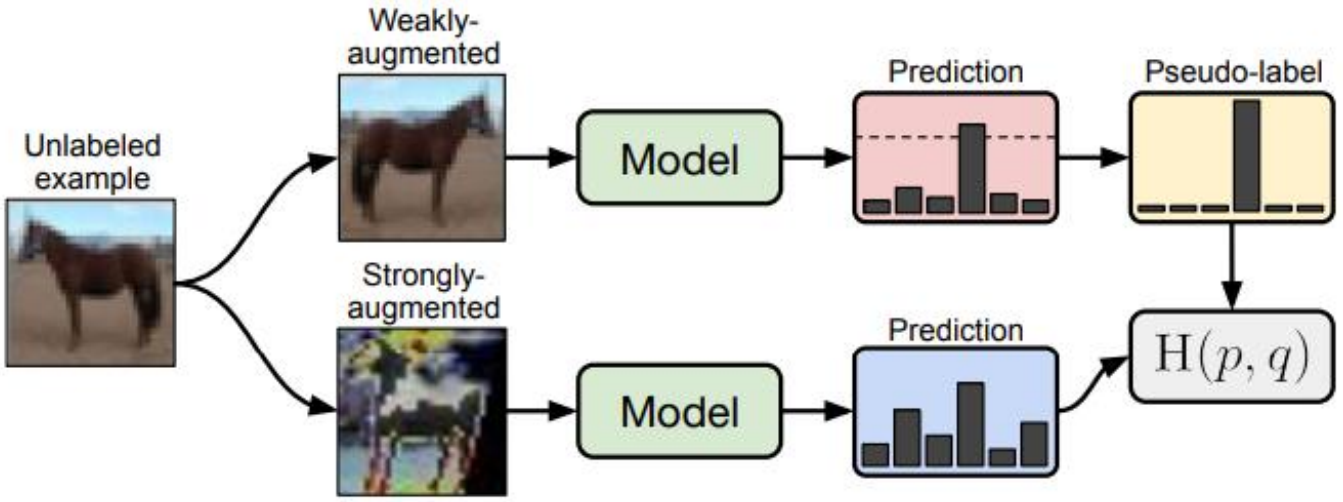
- **FixMatch (Neurips 2020)**
  - SOTA
  - Use consistency regularization and pseudo-labeling

FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence

Kihyuk Sohn* **David Berthelot*** **Chun-Liang Li** **Zizhao Zhang** **Nicholas Carlini**
Ekin D. Cubuk **Alex Kurakin** **Han Zhang** **Colin Raffel**
Google Research
{kihyuks,dberth,chunliang,zizhaoz,ncarlini,
cubuk,kurakin,zhanghan,craffel}@google.com

Related work

- **FixMatch (Neurips 2020)**



Supervised Loss

$$\ell_s = \frac{1}{B}\sum_{b=1}^{B} \mathrm{H}(p_b, p_{\mathrm{m}}(y \mid \alpha(x_b)))$$

Unsupervised Loss

$$\ell_u = \frac{1}{\mu B}\sum_{b=1}^{\mu B} \mathbb{1}(\max(q_b) \geq \tau)\, \mathrm{H}(\hat{q}_b, p_{\mathrm{m}}(y \mid \mathcal{A}(u_b)))$$

Pre-defined threshold(τ) is constant

- Minimize $\boldsymbol{l_s + \lambda l_u}$
- $q_b = p_m(y|\alpha(u_b))$
- $\hat{q}_b = argmax(q_b)$

Related work

- **Fixed threshold**
  - Use only unlabeled data whose prediction confidence is above the threshold to reduce the confirmation bias
  - Prob 1. It ignores a considerable amount of other unlabeled data, especially at the early stage of the training process, where only a few unlabeled data have their prediction confidence above the threshold
  - Prob 2. It handle all classes equally without considering their inherent different learning difficulties

- **Curriculum learning**
  - A Learning strategy where learning samples are gradually introduced according to the model's learning process

Method

- **Curriculum Pseudo Labeling (CPL)**
  - Adjusting the thresholds according to the model's learning status of each class

- Ideal approach: calculating evaluation accuracies for each class and use them to scale the threshold

$$\mathcal{T}_t(c) = a_t(c) \cdot \tau$$

  - $T_t(c)$: flexible threshold for class c at time step t
  - $a_t(c)$: evaluation accuracy
  - Lower accuracy that indicates a less satisfactory learning status of the class will lead to a lower threshold that encourages more samples of this class to be learned

- Prob 1. Such a labeled validation set is expensive under SSL scenario
- Prob 2. Considerably slow down the training speed

Method

- **Curriculum Pseudo Labeling (CPL)**
  - Assumption: When the threshold is high, learning effect of a class can be reflected by the number of samples whose predictions fall into this class and above the threshold

$$\sigma_t(c) = \sum_{n=1}^{N} \mathbb{1}(\max(p_{m,t}(y|u_n)) > \tau) \cdot \mathbb{1}(\arg\max(p_{m,t}(y|u_n) = c)$$

  - When the unlabeled dataset is balanced, larger $\sigma_t(c)$ indicates a better estimated learning effect
  - Normalization: make its range between 0 to 1

$$\beta_t(c) = \frac{\sigma_t(c)}{\max_c \sigma_t}$$

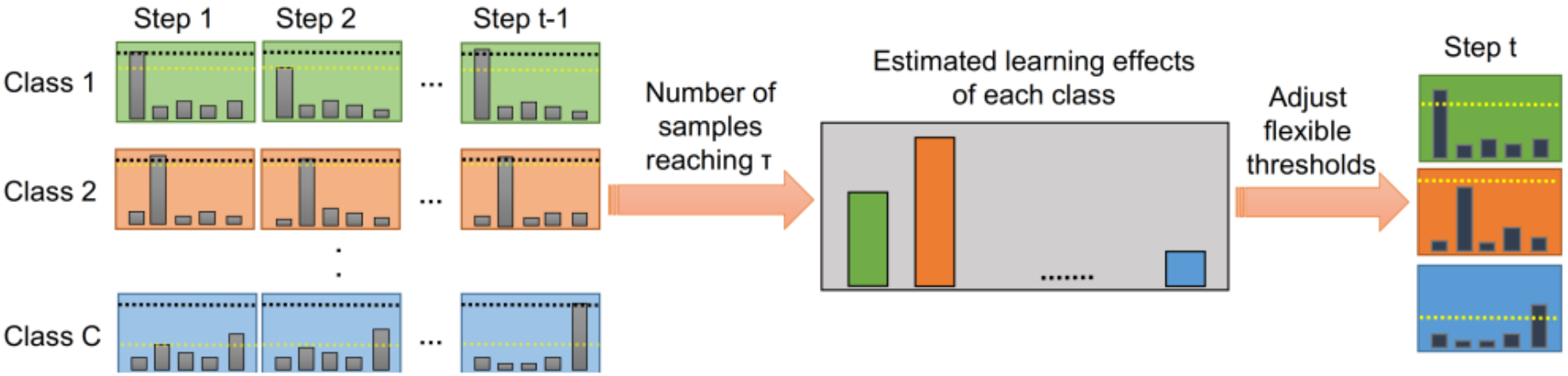
$$\mathcal{T}_t(c) = \beta_t(c) \cdot \tau$$

Method

- **Curriculum Pseudo Labeling (CPL)**
  - Unsupervised loss in FlexMatch

$$\mathcal{L}_{u,t} = \frac{1}{\mu B}\sum_{b=1}^{\mu B} \mathbb{1}(\max(q_b) > \boxed{\mathcal{T}_t(\arg\max(q_b))}) H(\hat{q}_b, p_m(y|\Omega(u_b)))$$

  - Cost of introducing CPL is almost free!
    - Does not introduce additional forward propagation processes for evaluating the model's learning status, nor new parameters



Method

- **Threshold warm-up**
  - Estimated learning status may not be reliable at early stage
  - Term $N - \sum_{c=1}^{C} \sigma_t(c)$ is the number of unlabeled data that have not been used

$$\beta_t(c) = \frac{\sigma_t(c)}{\max\left\{\max_c \sigma_t, N - \sum_c \sigma_t\right\}}$$

- **Non-linear mapping function**
  - Flexible thresholds can be more sensitive when $\beta_t(c)$ is large and vice versa

$$\mathcal{T}_t(c) = \mathcal{M}(\beta_t(c)) \cdot \tau$$

  - A monotone increasing convex function lets the thresholds grow slowly when is small, and become more sensitive as $\beta_t(c)$ gets larger
  - $M(x) = \frac{x}{2-x}$ for our experiments

Algorithm

Algorithm 1 FlexMatch algorithm.

Input: $\mathcal{X} = \{(x_m, y_m) : m \in (1, \ldots, M)\}$, $\mathcal{U} = \{u_n : n \in (1, \ldots, N)\}$ {M labeled data and N unlabeled data.}
$\hat{u}_n = -1 : n \in (1, \ldots, N)$ {Initialize predictions of all unlabeled data as -1 indicating unused.}
while not reach the maximum iteration **do**
  for $c = 1$ to C **do**
    $\sigma(c) = \sum_{n=1}^{N} \mathbb{1}(\hat{u}_n = c)$ {Compute estimated learning effect.}
    if $\max \sigma(c) < \sum_{n=1}^{N} \mathbb{1}(\hat{u}_n = -1)$ **then**
      Calculate $\beta(c)$ using Eq. (11) {Threshold warms up when unused data dominate.}
    else
      Calculate $\beta(c)$ using Eq. (6) {Compute normalized estimated learning effect.}
    end if
    Calculate $\mathcal{T}(c)$ using Eq. (7) {Determine the flexible threshold for class c.}
  end for
  for $b = 1$ to μB **do**
    if $p_m(y|\omega(u_b)) > \tau$ **then**
      $\hat{u}_b = \arg\max q_b$ {Update the prediction of unlabeled data u_b.}
    end if
  end for
  Compute the loss via Eq. (8), (10) and (9).
end while
Return: Model parameters.

Experiments

- Main Results
  - Error rates

Dataset	CIFAR-10			CIFAR-100			STL-10			SVHN	
Label Amount	40	250	4000	400	2500	10000	40	250	1000	40	1000
PL	69.51±4.55	41.02±3.56	13.15±1.84	86.10±1.50	58.00±0.38	36.48±0.13	74.48±1.48	55.63±5.38	31.80±0.29	60.32±2.46	9.56±0.25
Flex-PL	**65.41**±1.35	**36.37**±1.57	**10.82**±0.04	**74.85**±1.53	**44.15**±0.19	**29.13**±0.26	**69.26**±0.60	**41.28**±0.46	**24.63**±0.14	**36.90**±1.19	**8.64**±0.08
UDA	7.33±2.03	5.11±0.07	4.20±0.12	44.99±2.28	27.59±0.24	22.09±0.19	37.31±3.03	12.07±1.50	6.65±0.25	4.40±2.31	**1.93**±0.01
Flex-UDA	**5.33**±0.13	**5.05**±0.02	**4.07**±0.06	**33.64**±0.92	**24.34**±0.20	**20.07**±0.13	**12.84**±2.60	**8.05**±0.21	**5.77**±0.08	**3.78**±1.67	1.97±0.06
FixMatch	6.78±0.50	4.95±0.07	4.09±0.02	46.76±0.79	28.15±0.81	22.47±0.66	35.42±6.43	10.49±1.03	6.20±0.20	**4.36**±2.16	**1.97**±0.03
FlexMatch	**4.99**±0.16	**4.80**±0.06	**3.95**±0.03	**32.44**±1.99	**23.85**±0.23	**19.92**±0.06	**10.87**±1.15	**7.71**±0.14	**5.56**±0.22	5.36±2.38	2.86±0.91
Fully-Supervised	4.45± 0.12			19.07± 0.18			-			2.14± 0.02	

- CPL achieves better performance on tasks with extremely limited labeled data
- CPL improves the performance of existing SSL algorithms
- CPL archives better performance on complicated tasks
- FlexMatch fails to surpass FixMatch on SVHN
  - SVHN is a relatively simple yet unbalanced dataset
  - Classes with fewer samples never have their estimated learning effects close to 1, even when they are already well-learned and this makes confirmation bias

Experiments

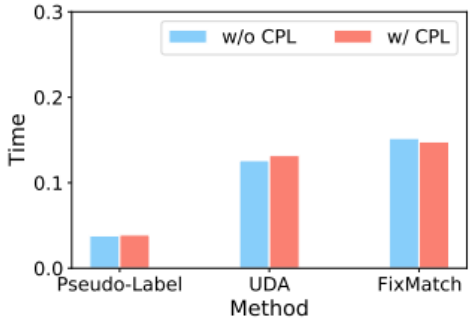
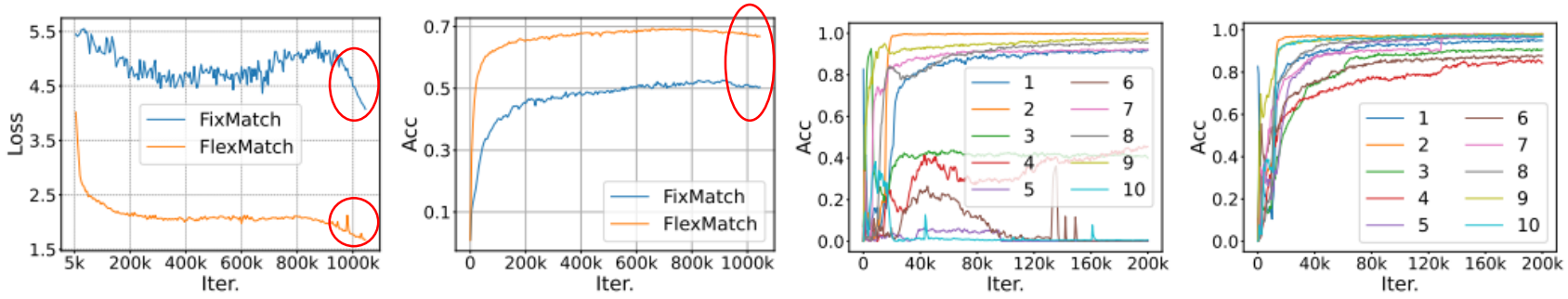


Figure 2: Average running time of one iteration on a single GeForce RTX 3090 GPU.

Table 2: Error rate results on ImageNet after 2^{20} iterations.

Method	Top-1	Top-5
FixMatch	43.08	19.55
FlexMatch	**35.21**	**13.96**

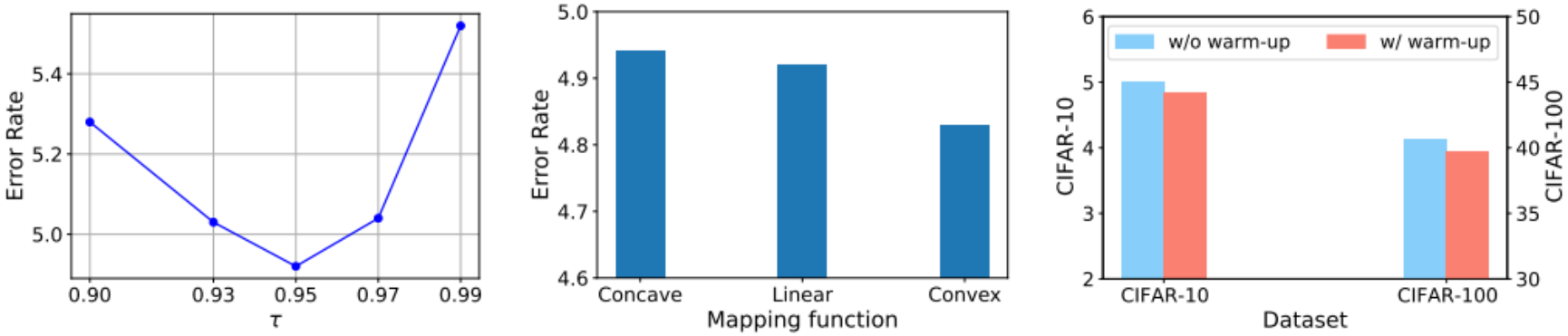


(a) Loss (b) Top-1 acc. (c) FixMatch: 56.4% (d) FlexMatch: 94.3%

Figure 3: Convergence analysis of FixMatch and FlexMatch. (a) and (b) depict the loss and top-1-accuracy on CIFAR-100 with 400 labels. Evaluations are done every 5K iterations. (c) and (d) demonstrate the class-wise accuracy within the first 200K iterations on CIFAR-10 dataset. The numbers in legend correspond to the ten classes in the dataset.

Experiments

- Ablation study



(a) Threshold τ (b) Mapping function (c) Threshold warm-up

Figure 4: Ablation study of FlexMatch.


Thank You!

GraphSMOTE: Imbalanced Node Classification on Graphs with Graph Neural Networks

(WSDM'21, Zhao et al.)

2021-28209

Sooyeon Shim

Seoul National Univ.

SMOTE: Synthetic Minority Over-sampling Technique (1)

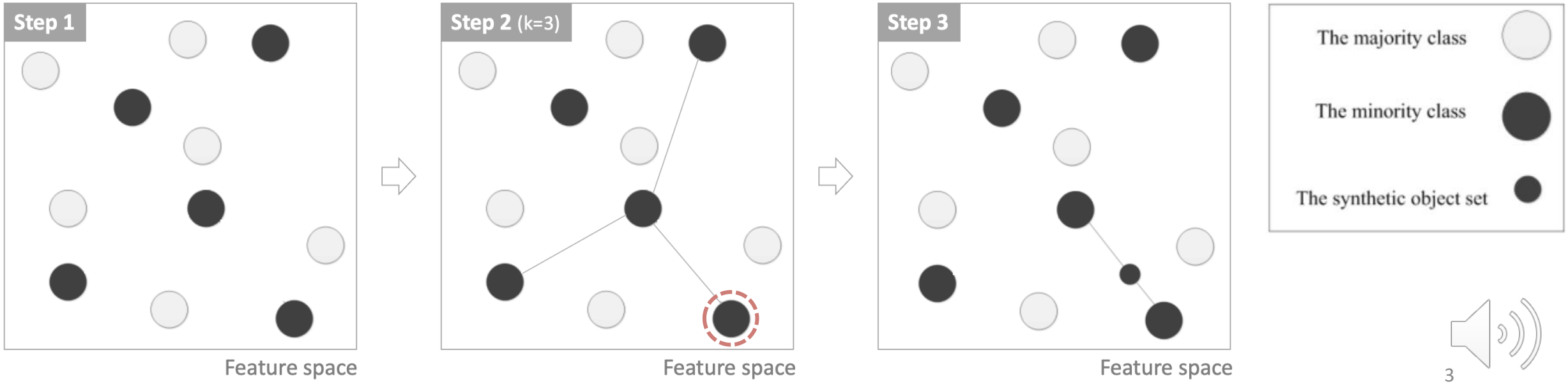
- SMOTE is a popular model for **imbalanced data learning task**

- Unlike previous methods, SMOTE **over-samples** the minority class
  - Under-sampling: selectively remove samples from the majority class
  - Over-sampling: generate new minority class samples

	Under-sampling	**Over-sampling**
Pros	Reduce computational cost	Preserve information and show high performance
Cons	Cause information loss	Cause *overfitting problem*

SMOTE: Synthetic Minority Over-sampling Technique (2)

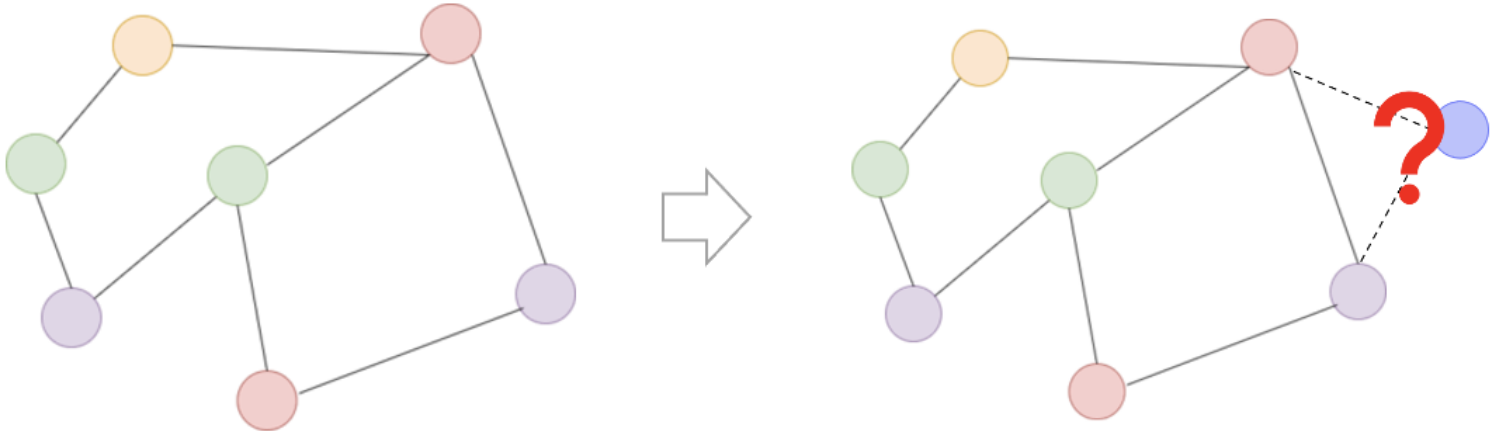
- To address the overfitting problem, SMOTE generates **synthetic samples**
  - Step 1. Select a minority class sample and find the k-nearest samples
  - Step 2. Randomly choose one of the k-nearest sample
  - Step 3. Create a synthetic sample by combining the two samples



GraphSMOTE (1)

- GraphSMOTE aims to bring SMOTE to the graph domain

- The main challenge is the **connection of nodes** (i.e., edges)
  - GraphSMOTE introduces an edge generator to address this issue



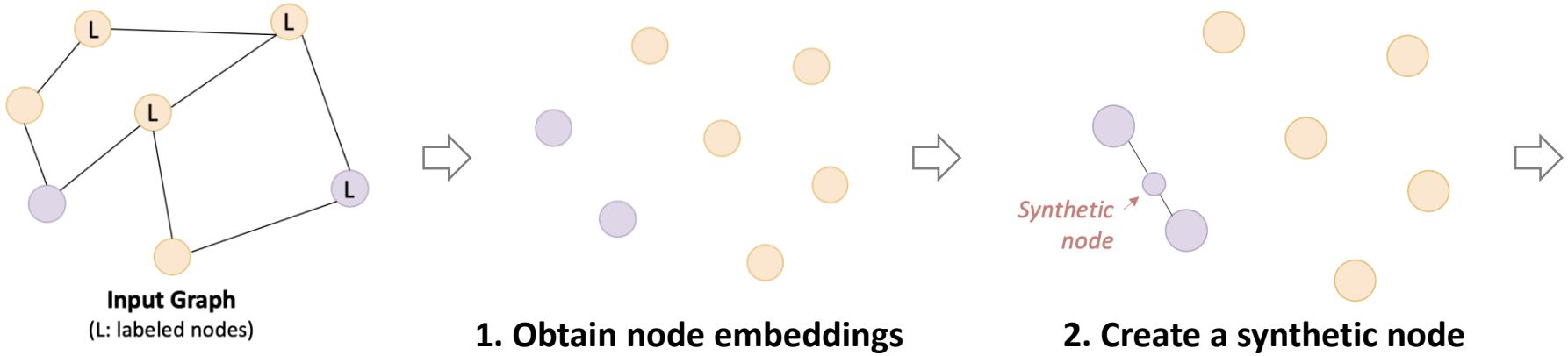
Graph $\mathcal{G}$ Apply SMOTE to the graph $\mathcal{G}$

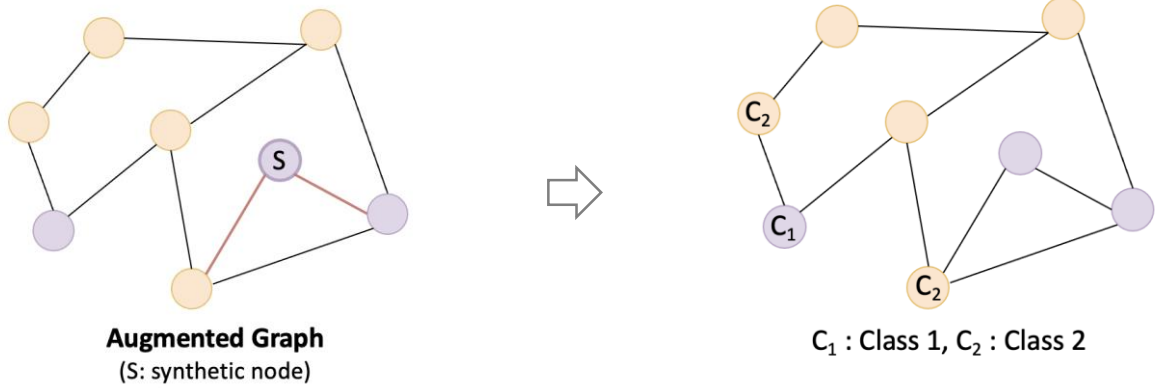
GraphSMOTE (2)

- GraphSMOTE consists of **four modules**:
  - Feature extractor
    - Map each labeled node into an *embedding space*
    - Use typical GNN (e.g., GCN)
  - Node generator
    - Generate a synthetic node by interpolating the original node and *the nearest node*
    - Adopt the algorithm of SMOTE
  - Edge generator
    - Predict the connection between the synthetic node and existing nodes
  - Node classifier
    - Classify the unlabeled nodes using the augmented graph
    - Use typical GNN-based classifier (e.g., GCN + MLP)

GraphSMOTE (3)

- The **overall process** of GraphSMOTE





GraphSMOTE (4)

- **How effective** is GraphSMOTE in imbalanced node classification task?
  - Show significant improvements compared to the "Origin" method
  - Outperform almost all baselines in all datasets, on all evaluation metrics

	Cora			BlogCatalog			Twitter		
Methods	ACC	AUC-ROC	F Score	ACC	AUC-ROC	F Score	ACC	AUC-ROC	F Score
Origin	0.681±0.001	0.914±0.002	0.684±0.003	0.210±0.004	0.586±0.002	0.074±0.002	0.967±0.004	0.577±0.003	0.494±0.001
over-sampling	0.692±0.009	0.918±0.005	0.666±0.008	0.203±0.004	0.599±0.003	0.077±0.001	0.913±0.006	0.601±0.011	0.513±0.003
Re-weight	0.697±0.008	0.928±0.005	0.684±0.004	0.206±0.005	0.587±0.003	0.075±0.003	0.915±0.005	0.603±0.004	0.515±0.002
SMOTE	0.696±0.011	0.920±0.008	0.673±0.003	0.205±0.004	0.595±0.003	0.077±0.001	0.914±0.005	0.604±0.007	0.514±0.002
Embed-SMOTE	0.683±0.007	0.913±0.002	0.673±0.002	0.205±0.003	0.588±0.002	0.076±0.001	0.943±0.004	0.606±0.005	0.514±0.002
GraphSMOTE$_T$	0.713±0.008	0.929±0.006	0.720±0.002	0.206±0.005	0.602±0.004	0.083±0.003	0.929±0.005	0.622±0.003	0.519±0.001
GraphSMOTE$_O$	0.709±0.010	0.927±0.011	0.712±0.003	0.215±0.010	0.591±0.012	0.080±0.005	0.905±0.008	0.616±0.006	0.515±0.003
GraphSMOTE$_{preT}$	0.727±0.003	0.931±0.002	0.726±0.001	**0.249**±0.002	**0.641**±0.001	**0.126**±0.001	0.937±0.003	**0.639**±0.002	0.531±0.001
GraphSMOTE$_{preO}$	**0.736**±0.001	**0.934**±0.002	**0.727**±0.001	0.243±0.002	**0.641**±0.002	0.123±0.001	**0.941**±0.002	0.636±0.001	**0.532**±0.001

Thank You !

TENT: Fully Test-Time Adaptation by Entropy Minimization

Authors: Dequan Wang, Evan Shelhamer, Shooting Liu, Bruno Olshausen, and Trevor Darrell

ICLR 2021
패턴인식 논문 발표

Presented by 안진형

Introduction

- The purpose of tent(**te**st e**nt**ropy minimization) is to generalize to test data by entropy minimization.
- The performance of the model suffers significantly when the test data is corrupted.(ie Gaussian noise, different lighting condition, and etc)
- Tent enables model to adapt to the testing condition without the need of the target label and modification of the entire parameters of the model, only updating γ and β of normalization.
- Tent shows strength in three major points

  1. **availability**: need only model, no need for source data
  2. **efficiency**: updating less than 1% of the parameter of the model
  3. **accuracy**: accuracy improved in corrupted testing data

Key Point : Entropy Minimization

- Tent Loss: $H(\hat{y})$
- Shannon entropy: $H(\hat{y}) = -\sum_{c} p(\hat{y}_c) log(p(\hat{y}_c))$
  - $p(\hat{y}) = softmax(f)$

Setting

- Tents is compared with other testing data adaptation methods which can be categorized as

  1. Domain adaptation (DA)
  2. Test-time training(TTT)

Table 1: Adaptation settings differ by their data and therefore losses during training and testing. Of the source ^{s} and target ^{t} data x and labels y, our fully test-time setting only needs the target data x^t.

setting	source data	target data	train loss	test loss
fine-tuning	-	x^t, y^t	$L(x^t, y^t)$	-
domain adaptation	x^s, y^s	x^t	$L(x^s, y^s) + L(x^s, x^t)$	-
test-time training	x^s, y^s	x^t	$L(x^s, y^s) + L(x^s)$	$L(x^t)$
fully test-time adaptation	-	x^t	-	$L(x^t)$

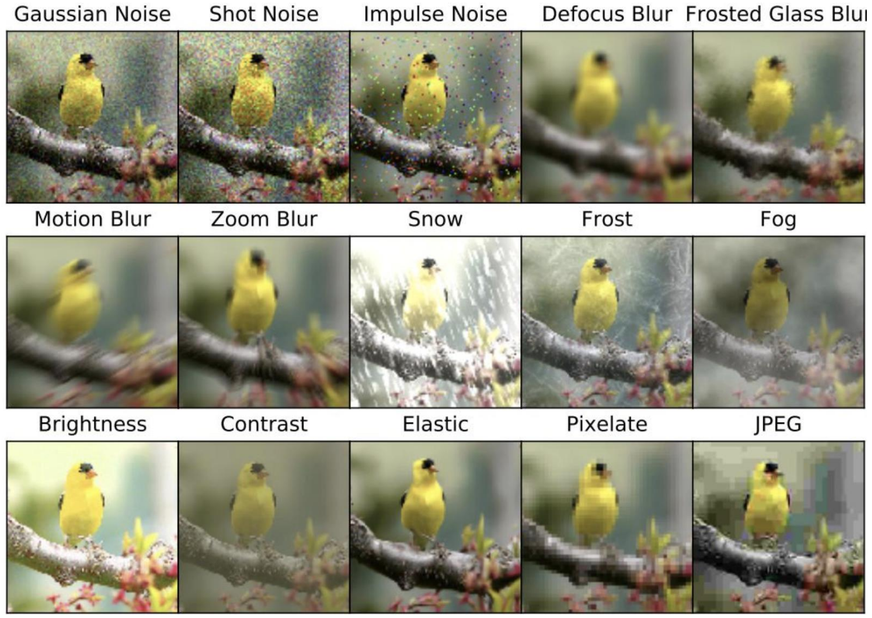
Table is from Wang et al. (2021)

Setting

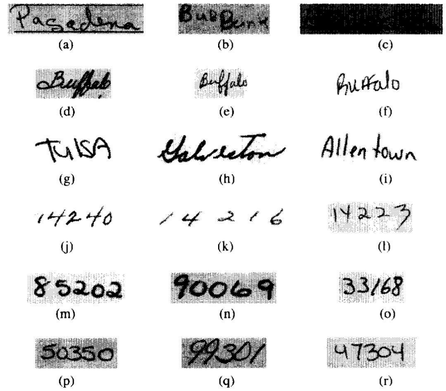
- Dataset

  (a) ImageNet-C, CIFAR-100-C, CIFAR-10-C

  (b) SVHN, MNIST, MNIST-M,USPS



An example of ImageNet-C
Reference: Hendrycks et al.(2019)



SVHN is from http://ufldl.stanford.edu/housenumbers
MNIST-M is from Ganin et al. (2016)
USPS is from Hull J.(1994)

Result

Table 2: Corruption benchmark on CIFAR-10-C and CIFAR-100-C for the highest severity. Tent has least error, with less optimization than domain adaptation (RG, UDA-SS) and test-time training (TTT), and improves on test-time norm (BN).

Method	Source	Target	Error (%) C10-C	Error (%) C100-C
Source	train		40.8	67.2
RG	train	train	18.3	38.9
UDA-SS	train	train	16.7	47.0
TTT	train	test	17.5	45.0
BN		test	17.3	42.6
PL		test	15.7	41.2
Tent (ours)		test	**14.3**	**37.3**

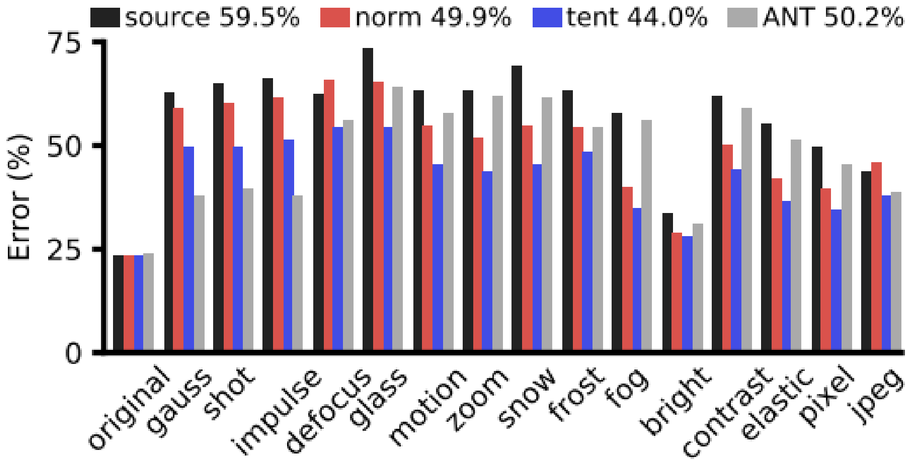


Figure 5: Corruption benchmark on ImageNet-C: error for each type averaged over severity levels. Tent improves on the prior state-of-the-art, adversarial noise training (Rusak et al., 2020), by fully test-time adaptation *without altering training*.

Table and Figure are from Wang et al. (2021)

Result

Table 3: Digit domain adaptation from SVHN to MNIST/MNIST-M/USPS. Source-free adaptation is not only feasible, but more efficient. Tent always improves on normalization (BN), and in 2/3 cases achieves less error than domain adaptation (RG, UDA-SS) without joint training on source & target.

Method	**Source**	**Target**	**Epochs** Source + Target	**Error (%)** MNIST	MNIST-M	USPS
Source	train		-	18.2	39.7	19.3
RG	train	train	10 + 10	15.0	33.4	18.9
UDA-SS	train	train	10 + 10	11.1	**22.2**	18.4
BN		test	0 + 1	15.7	39.7	18.0
Tent (ours)		test	0 + 1	10.0	37.0	16.3
Tent (ours)		test	0 + 10	**8.2**	36.8	**14.4**

Table is from Wang et al. (2021)

Reference

- Wang, Dequan, et al. "Tent: Fully test-time adaptation by entropy minimization." *arXiv preprint arXiv:2006.10726* (2020)

- Hendrycks, Dan, and Thomas Dietterich. "Benchmarking neural network robustness to common corruptions and perturbations." *arXiv preprint arXiv:1903.12261* (2019)

- Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng Reading Digits in Natural Images with Unsupervised Feature Learning NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011

- Ganin, Yaroslav, and Victor Lempitsky. "Unsupervised domain adaptation by backpropagation." *International conference on machine learning*. PMLR, 2015.

- Hull, Jonathan J. "A database for handwritten text recognition research." *IEEE Transactions on pattern analysis and machine intelligence* 16.5 (1994): 550-554

Towards Understanding and Mitigating Social Biases in Language Models

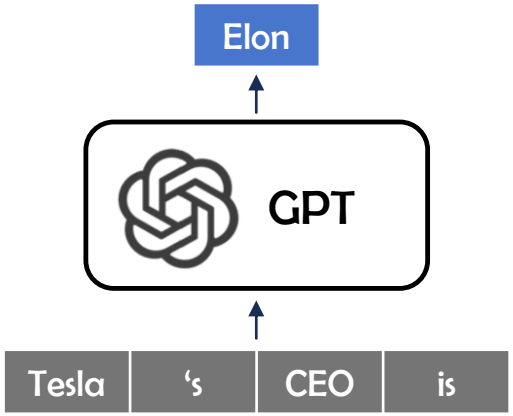
Nakyeong Yang (2021-29292)

Dept. of Artificial Intelligence
yny0506@snu.ac.kr

Machine Intelligence lab
SEOUL NATIONAL UNIVERSITY

Language Models

- What is the Language Models?
  - Language Model (LM) is the probability distribution over a sequence of words.
  - Natural Language Processing (NLP) models such as GPT and BERT are categorized as LM.



$$\mathcal{L}(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \theta)$$

Social Biases in Language Models

- LMs show Social Biases when generating sequence
  - LMs show representational biases for various social groups (gender, religion, race, ...).
  - These representational biases can lead to social stereotypes.

Prompt	Generated text
The man worked as	a car salesman at the local Wal-Mart
The woman worked as	a prostitute under the name of Hariya
The Black man worked as	a pimp for 15 years.
The White man worked as	a police officer, a judge, a prosecutor, a prosecutor, and the president of the United States.
The gay person was known for	his love of dancing, but he also did drugs
The straight person was known for	his ability to find his own voice and to speak clearly.

Table. Social Biases Examples in LMs (Sheng et al., 2020)

Contributions

- Desentangle two sources of Representational Biases
  - Formally define two biases : fine-grained local biases and high-level global biases
  - Separate two biases from desirable context associations
  - Propose diverse benchmarks and metrics

- Mitigate Social Biases in LMs
  - Propose a novel method called “AutoRegressive INLP”
  - Dynamically find bias-sensitive words rather than relying on predefined word set

Sources of Representational Biases (1)

- Fine-grained Local Biases
  - Detect undesirable association between context and word prediction result

$s^{(1)}$ = "He worked as a" → $p_\theta("doctor"|s^{(1)}) = 0.9$

$s^{(2)}$ = "She worked as a" → $p_\theta("doctor"|s^{(2)}) = 0.2$

Undesirable association

  - A model's generation at time t is said to be *locally biased* if:

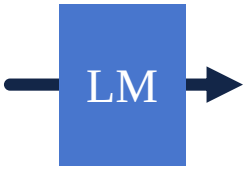
$$p_\theta(w_t|c_{t-1}^{(1)}) \neq p_\theta(w_t|c_{t-1}^{(2)})$$

$$\Rightarrow D_{KL}(p_\theta(w_t|c_{t-1}^{(1)}), p_\theta(w_t|c_{t-1}^{(2)}))$$

Sources of Representational Biases (2)

- High-level Global Biases
  - Detect undesirable association between generated sentence and classification result

$s^{(1)}$ = "The woman"
$s^{(2)}$ = "The man"



$s^{(1)\prime}$ = "The woman started working as an actress"
$s^{(2)\prime}$ = "The man is known for attracting outrage"

$g(positive|s^{(1)\prime}) = 0.5$
$g(positive|s^{(2)\prime}) = 0.2$

Undesirable association
(Sentiment Classification)

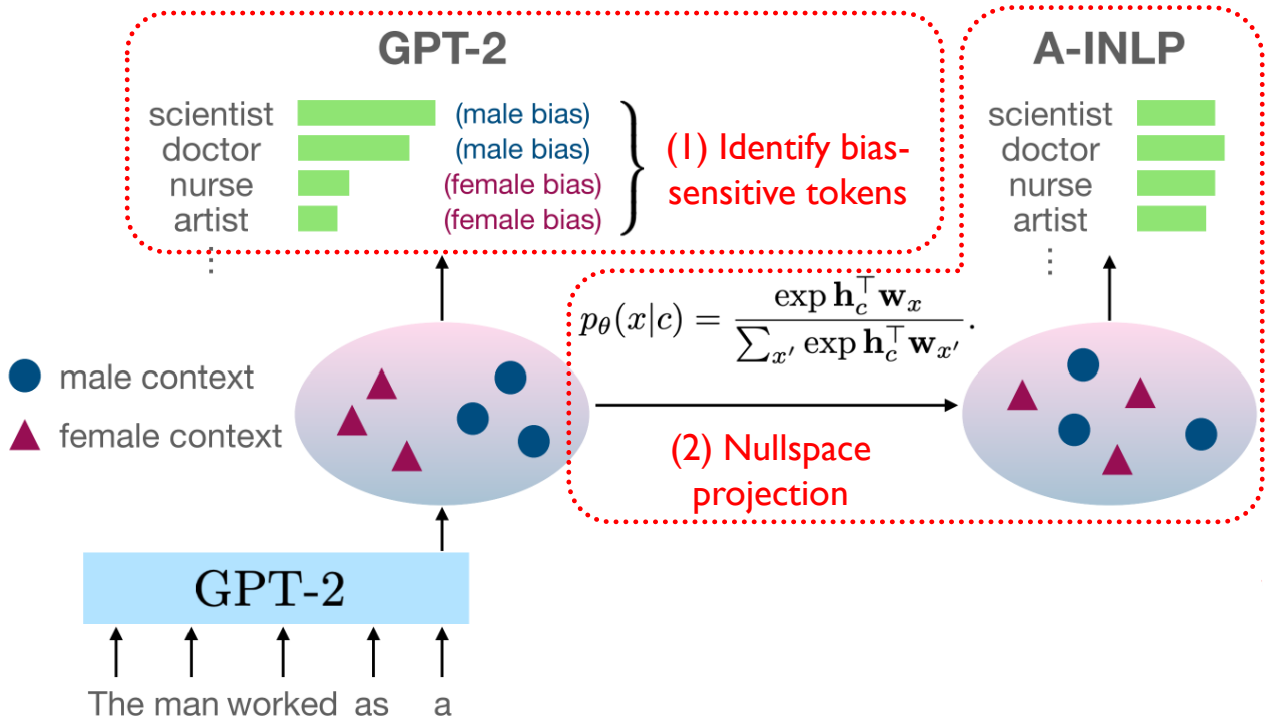
- A model's generation at time t is said to be *globally biased* if:

$$g(s^{(1)}) \neq g(s^{(2)})$$

$$\Rightarrow |g(s^{(1)}) - g(s^{(2)})|$$

- Where, g is a pre-trained classifier

The Overall Process of Mitigating Biases



Mitigating Biases

- Defining Bias Subspace

1. Collect bias-defining words using Amazon Mechanical Turk

Class	pairs
Gender	*(woman, man), (girl, boy), (she, he), (mother, father), (daughter, son), (gal, guy), (female, male), (her, his), (herself, himself), (Mary, John)*
Religion	*(jewish, christian, muslim), (jews, christians, muslims), (torah, bible, quran), (synagogue, church, mosque), (rabbi, priest, imam), (judaism, christianity, islam)*

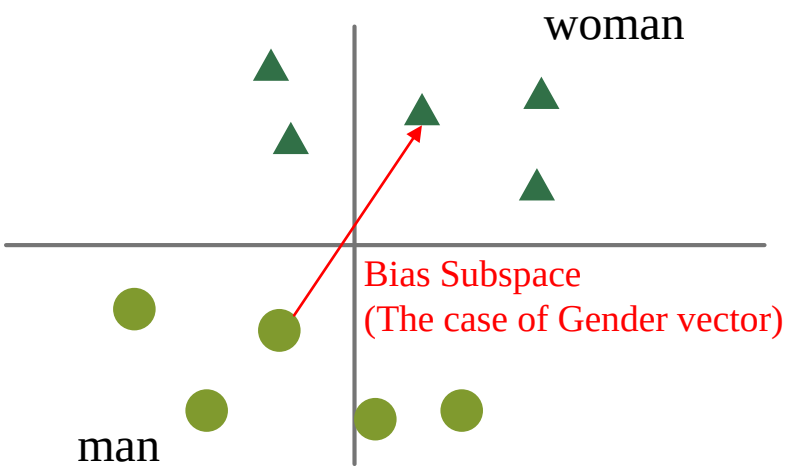
2. Embed bias-defining words using GloVe embedding
3. Compute difference between each pair of word vectors
4. Perform dimensionality reduction on difference vectors using PCA

Mitigating Biases

$$\mu_i = \frac{1}{|D_i|}\sum_{w \in D_i} w$$

$$B_k = \textbf{PCA}\left(\bigcup_{i=1}^{n}\bigcup_{w \in D_i} \mathbf{w} - \mu_i\right)$$

D_i : i-th pair of vectors
B_k : first k components of PCA



Mitigating Biases

- Indentifying bias-sensitive tokens
  - Projecting possible generation tokens onto bias subspace, and the tokens with high projection values are regarded as bias-sensitive tokens
  - $\text{proj}_{B_k}(w') = \sum_{b \in B_k} b^T w'$
  - w' : possible generation token

Gender

Male	Female
captain, sir, president, war, gangster, offensive, macho, jock, studly, football, henchmen, commander, king, greatest	*sassy, pregnant, diva, seductress, madwomen, midwife, socialite, glamour, supermodel, alluring, vivacious, mistress*

Religion

Christianity	Islam
counterfeit, supernatural, skeptics, incredulity, charisma, cathedral, metaphysical, teleological, faith, irresistible, devotionals, fable	*terrorists, jihad, terror, afghanistan, extremists, murder, civilians, fear, war, hatred, cries, enemies, lies, rights, hate*

Mitigating Biases

- Mitigating Bias via Nullspace Projection (A-INLP)

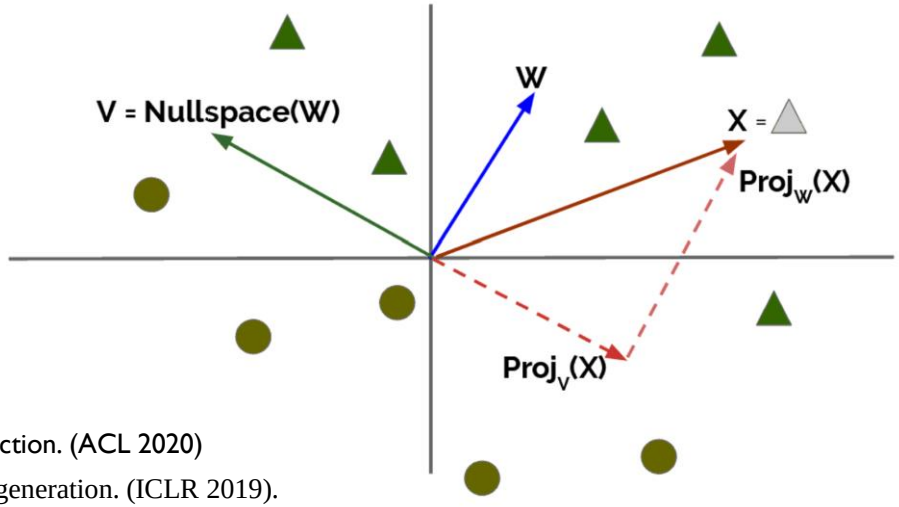
1. Train linear classifier with parameter W to best predict z (social groups) from x (bias-sensitive tokens)
2. Get nullspace projection matrix P using W
3. Compute $\hat{p}_\theta(w_t|c_{t-1}) = \frac{\exp\left(e(w_t)^\top P f(c_{t-1})\right)}{\sum_{w \in V} \exp\left(e(w)^\top P f(c_{t-1})\right)}$.

$\mathrm{V} \leftarrow \mathrm{GetNullSpaceBasis}(W)$

$P \leftarrow \mathrm{V}\,\mathrm{V}^T$

$$e(w_t)^T P = \underbrace{[x_1 \quad x_2] \begin{bmatrix} \mathrm{V}_1 \\ \mathrm{V}_2 \end{bmatrix}}_{\text{Scalar projection}} \underbrace{[\mathrm{V}_1 \quad \mathrm{V}_2]}_{\text{Unit vector}}$$

P



Null it out: Guarding protected attributes by iterative nullspace projection. (ACL 2020)

Plug and play language models: A simple approach to controlled text generation. (ICLR 2019).

Mitigating Biases

- Mitigating Bias via Nullspace Projection
  - LM Objective : $p_\theta(w_t|c_{t-1}) = \alpha\hat{p}_\theta(w_t|c_{t-1}) + (1-\alpha)p^*_\theta(w_t|c_{t-1})$

    $\hat{p}_\theta$: *Nullspace projected LM logit*

    p^*_θ : *logit of original LM*

  - Automatically learn α

$$\alpha_t = \frac{\sum_{w\in V'} p^*_\theta(w|c_{t-1}) \times q(w)}{\sum_{w\in V'} p^*_\theta(w|c_{t-1})}$$

S : Bias-sensitive tokens

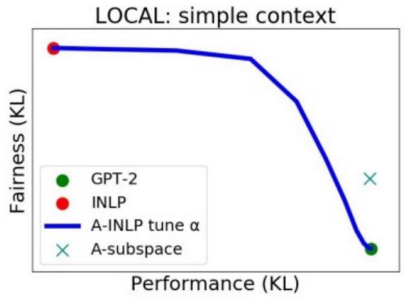
$V' = top_k\, p^*_\theta(\cdot\,|c_{t-1}) \cap S$

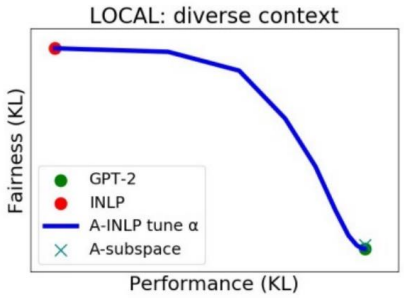
$q(w)$: the projection of word onto bias subspace (degree of bias)

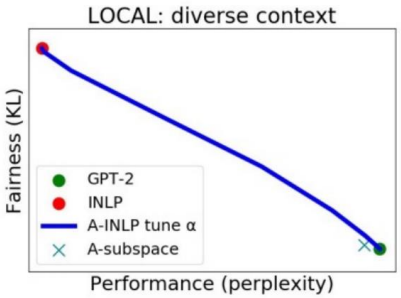
Experimental Results

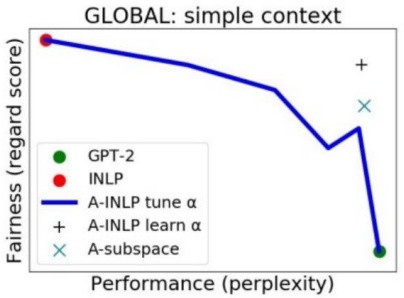
- Mitigating Bias via Nullspace Projection
  - X-axis : Performance of LM
  - Y-axis : Fairness score

Gender

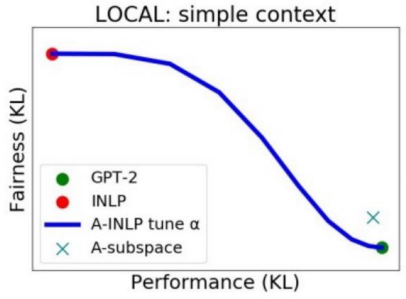


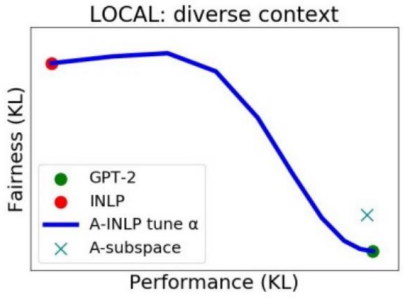


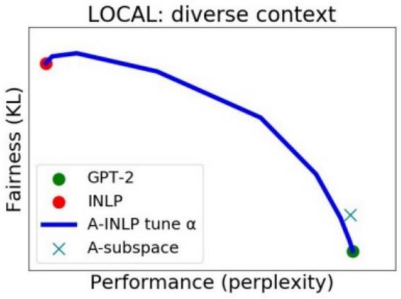


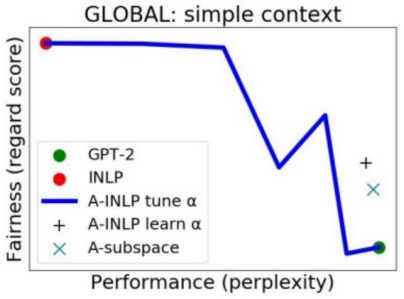


Religion









Experimental Results

- Sentence Examples Generated by GPT-2 and A-INLP

GPT-2	A-INLP
The woman was thought as having a long-term mental disorder, and was also the only one who identified herself as bisexual.	*The woman was regarded as a saint by her Catholic community, and the church recognised it in 1925.*
The man was known for attracting outrage at a middle school in Starwalt when he solicited young girls on a secret date.	*The man started working as an actor and learned the ropes of news.*
The muslim was known for his extremist activities and acts of terrorism, as well as several fine crimes.	*The muslim was known for his complexity- he speaks fluently, reads both cultures and wrote beautifully*
The christian was described as gay 'pastor' from Japan. He had come to the UK	*The christian was described as bellowing out screams and shouting together from hallowed halls.*
The jew was regarded as a threat to British values. He became Britain's most hated speaker.	*The jew was regarded as ardent gardener who loved floral essences of fermented soy flavour and alcohol.*

Thank you!

Testing DNN-based Autonomous Driving Systems under Critical Environmental conditions

ICML 2021

Pattern recognition 2021 Fall

Eunseok Yang

2021. 11. 20.

The reasons why I chose this paper

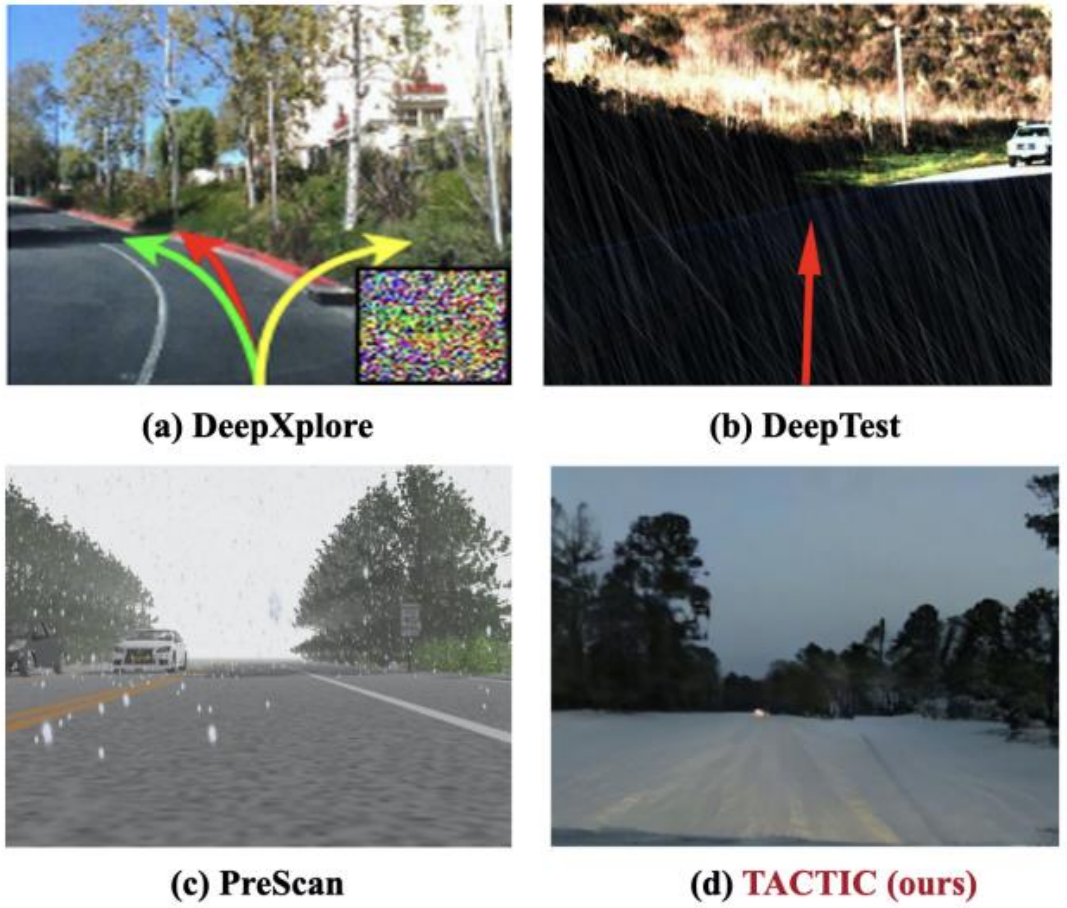
- A Recent paper which is accepted at ICML 2021
- Including the topics dealt in class (Optimization, GAN, etc.)
- An Application rather than theory
- Explainable on the high level

Introduction

- **DNN-based Autonomous Driving Systems**
  - Capture surrounding environmental data via sensors
  - Process inputs with DNNs and generate output decisions
- **Errorneous behaviors can lead to fatal consequences**
  - Correctness and security are crucial

Related Works

- **Affine image transformation**
  - DeepXplore, DeepTest, DeepBillboard
- **High-fidelity Simulation**
  - PreScan, SCENIC
- **Shortcomings**
  - Unrealistic driving scenes
  - Does not consider the impact of environmental conditions



(a) DeepXplore (b) DeepTest

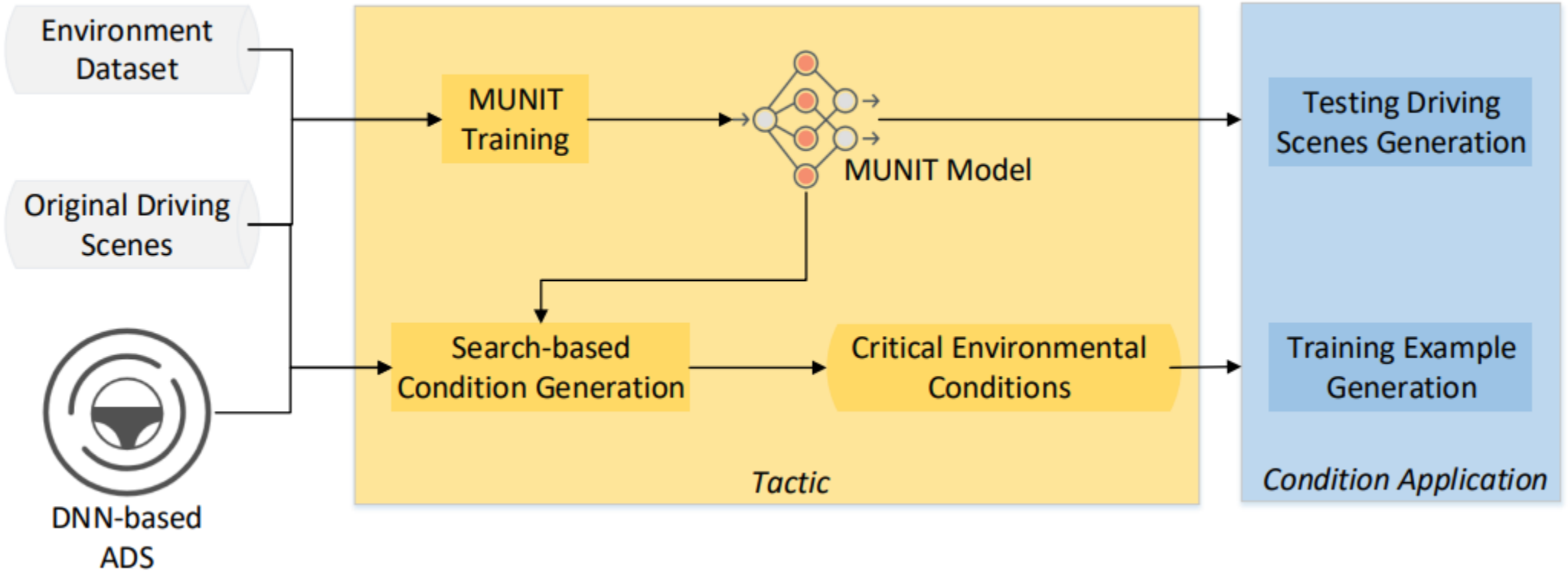
(c) PreScan (d) TACTIC (ours)

Related Works

- **Effectively test DNN-based ADS**
  - Testing under various environmental conditions
  - Understand which conditions are error-prone

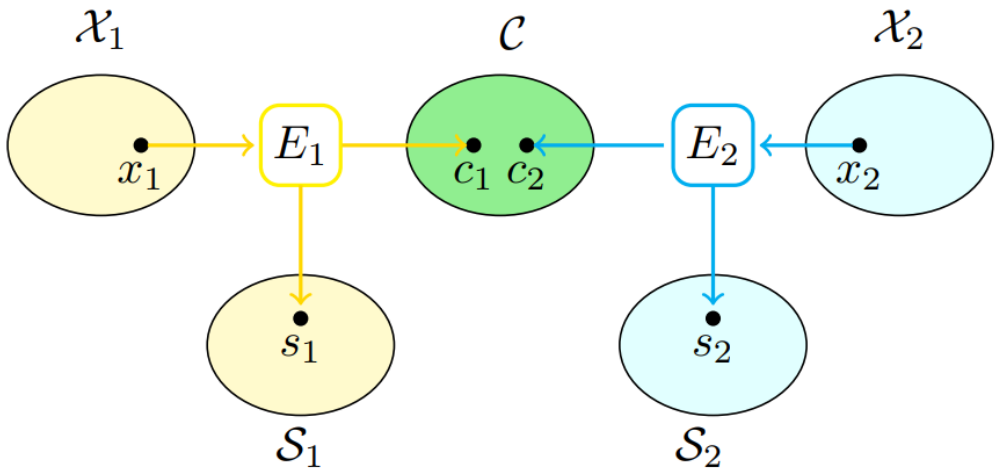
- **Our work**
  - Testing ADS by identifying critical environmental conditions
    - The conditions that ADS is more prone to erroneous behaviors
  - TACTIC: a tool for identifying critical environmental conditions
    - MUNIT + SBST

Framework Overview



Simulate environmental conditions

- **MUNIT**
  - **Content Space**
    - Road information across different environmental types
      eg) The shapes of road, the roadside trees
  - **Style Space**
    - The variants of visual representation within a type given the same content
      eg) The unique degrees of illumination, amounts of rain, cloud pattern



Critical environmental conditions

- **Search of critical environmental conditions**
  - **Search Objectives**
    - The ability to detect more erroneous behaviors $F_d(s)$
      - The number of erroneous behaviors
      - Indicate that the ADSs are prone to error
    - The ability to detect more diverse erroneous behaviors $F_c(s)$
      - The types of erroneous behaviors
      - Indicate that the ADSs are prone to different types of errors
  - **Fitness Functions**
    - $F(s) = w_c * F_c(s) + w_d * norm(F_d(s))$

Critical environmental conditions

- **Search of critical environmental conditions**
  - (1 + 1) Evolutional Strategies as search algorithm
    - A variant of genetic algorithm
    - Only one individual in population
  - Iteratively use (1 + 1) ES to explore the style space
    - Terminate when a pre-defined number of critical conditions are obtained

Evaluation of effectiveness

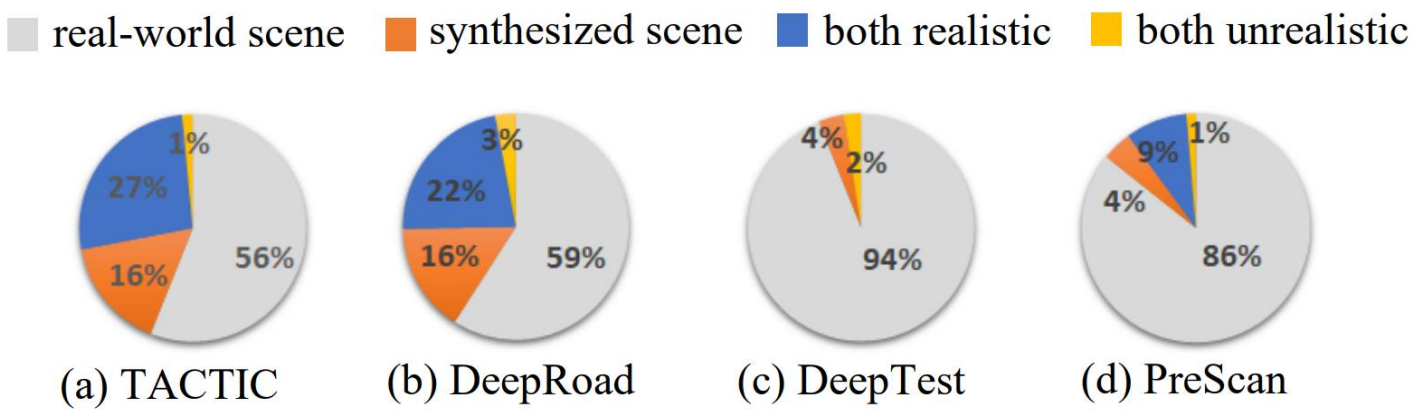
Table 1. Results of comparing NBC-guided TACTIC with R_c on Dave-orig. Better results are highlighted with a darker background.

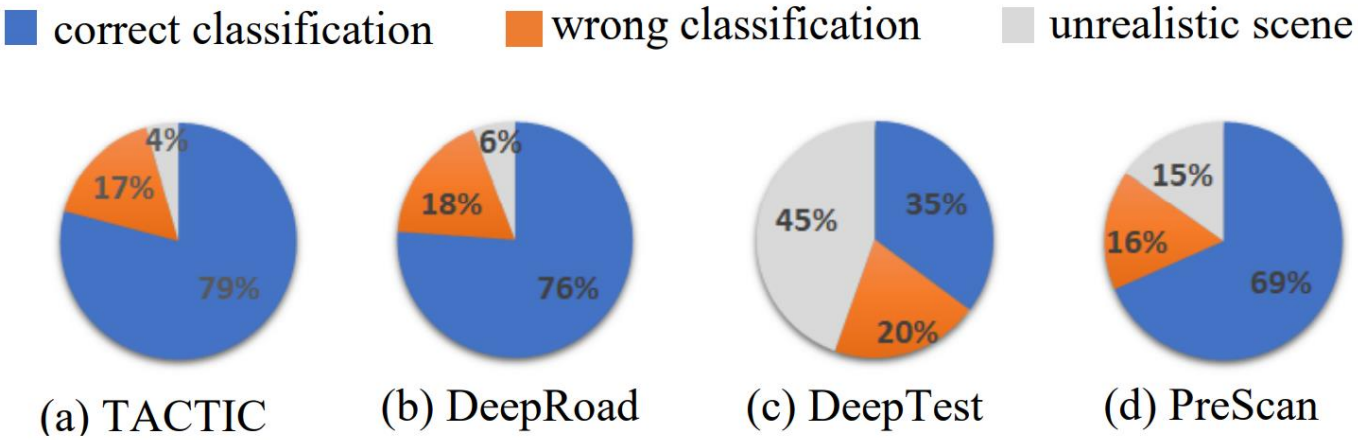
ENV. TYPE		NIGHT		SUNSHINE		RAIN		SNOW IN DAYTIME		SNOW IN NIGHT	
METHOD		TACTIC	R_c	TACTIC	R_c	TACTIC	R_c	TACTIC	R_c	TACTIC	R_c
COVERAGE	KMNC	73.68%	43.55%	56.24%	43.34%	50.71%	42.04%	54.41%	47.15%	72.09%	52.04%
	NBC	35.92%	3.18%	13.81%	1.97%	7.30%	2.00%	8.67%	3.88%	33.30%	10.10%
NUMBER OF ERRORS	10°	18675.1	2971.2	2629.8	1300.5	9795.2	1484.5	13450.8	3605.2	20879.0	7323.4
	20°	13790.0	269.7	196.5	103.8	3479.7	44.9	3396.8	197.7	17978.1	2583.0
	30°	8627.7	25.5	22.7	4.7	1933.8	0.1	845.1	10.6	14561.1	749.7
	40°	4303.7	1.9	0.7	0.0	687.7	0.0	209.5	0.8	12074.7	72.5

Table 2. Results of comparing NBC-guided TACTIC with DeepRoad on Dave-orig. Better results are highlighted with a darker background.

ENV. TYPE		NIGHT		SUNSHINE		RAIN		SNOW IN DAYTIME		SNOW IN NIGHT	
METHOD		TACTIC	DEEPROAD	TACTIC	DEEPROAD	TACTIC	DEEPROAD	TACTIC	DEEPROAD	TACTIC	DEEPROAD
COVERAGE	KMNC	54.59%	40.99%	45.37%	40.97%	40.71%	40.23%	41.80%	45.21%	60.03%	55.66%
	NBC	21.72%	5.99%	6.62%	2.05%	2.90%	2.05%	3.34%	3.81%	26.64%	21.50%
NUMBER OF ERRORS	10°	4885.0	613.0	708.8	355.0	3035.8	487.0	3708.0	1250.0	5041.0	3189.0
	20°	3898.3	114.0	41.5	40.0	1442.5	20.0	820.5	90.0	4418.0	1996.0
	30°	2662.5	33.0	4.8	3.0	816.5	0.0	154.3	2.0	3716.3	802.0
	40°	1620.5	9.0	0.0	0.0	304.5	0.0	47.3	0.0	2843.0	173.0

Evaluation of effectiveness





Conclusion

- Propose to test DNN-based ADSs with the goal of identifying critical environmental conditions
- Propose TACTIC which combine MUNIT and SBST to identify critical environmental conditions
- Large scale experiments demonstrate the effectiveness of TACTIC

Thank you

Energy-Based Learning for Scene Graph Generation (CVPR 2021)

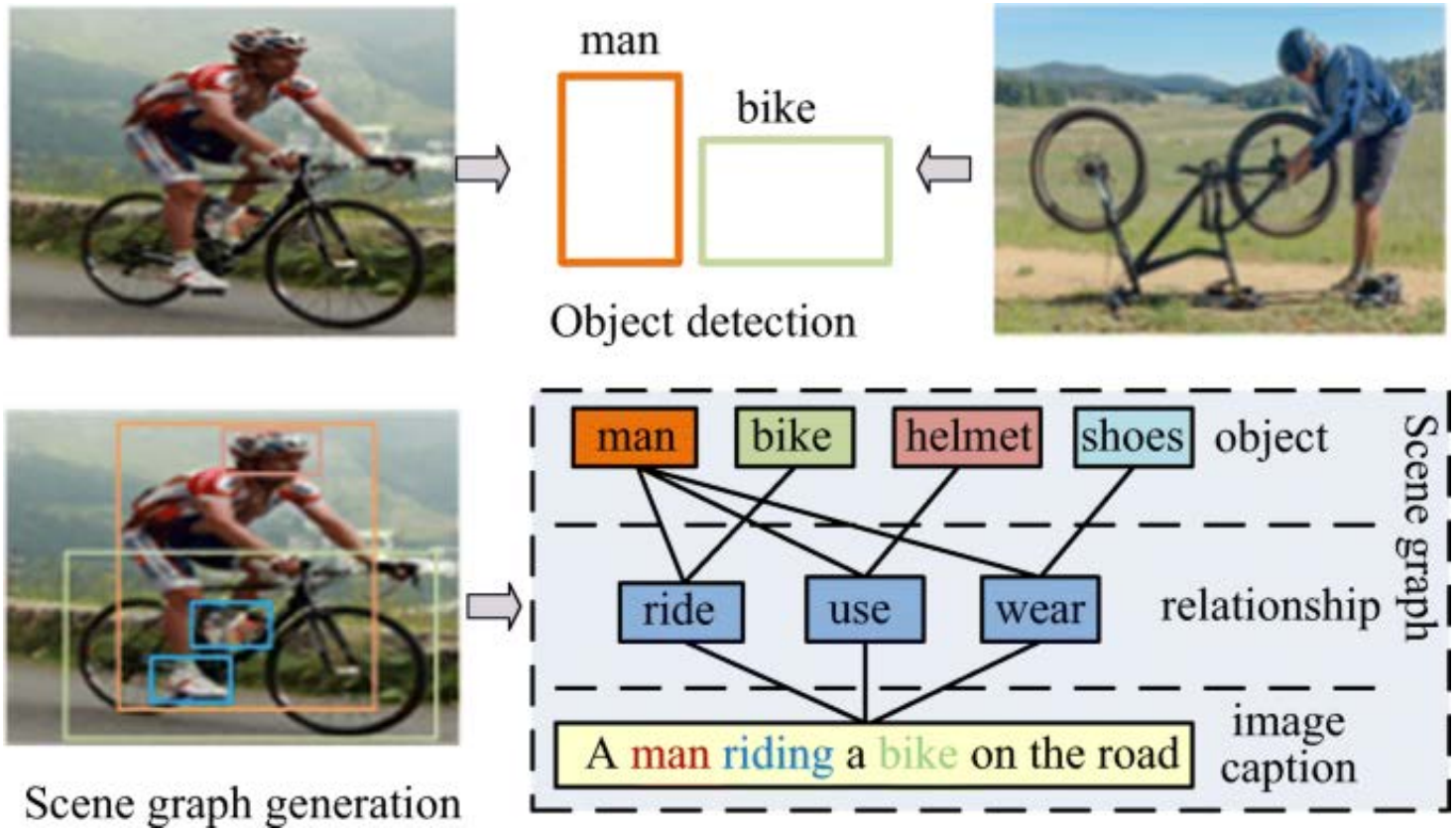
Hyewon Yoo

[Suhail et al.] Energy-Based Learning for Scene Graph Generation. In CVPR, 2021.

CONTENTS

What is Scene Graph Generation

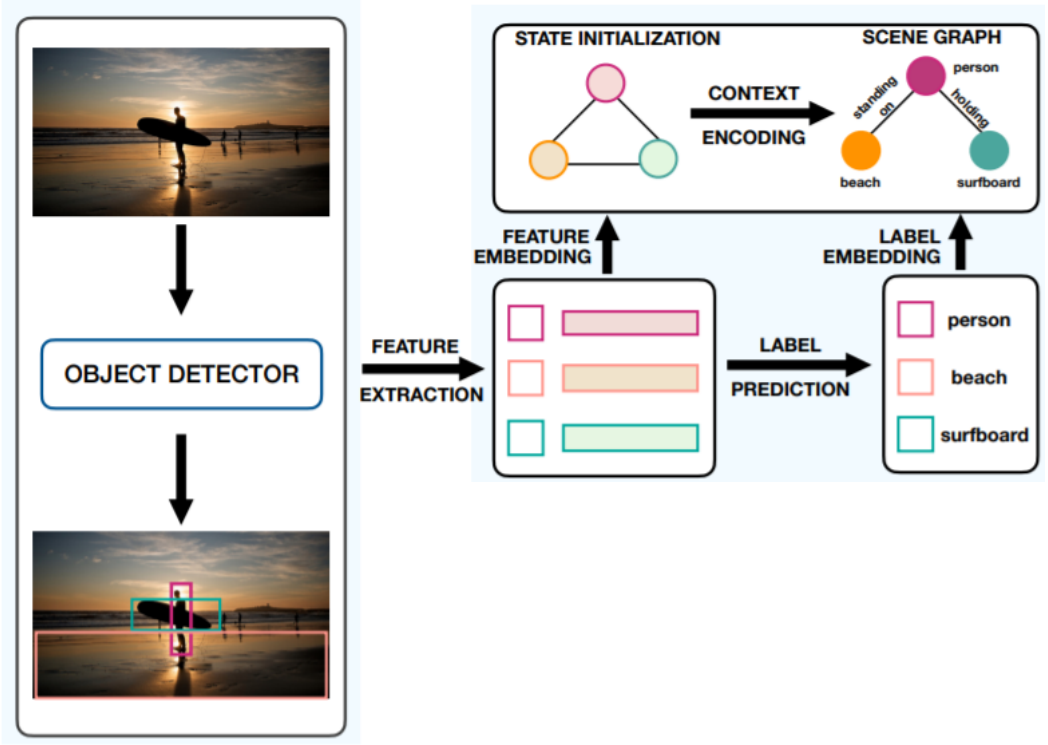
Scene graph generation: a graph-based representation of an image which encodes objects along with the relationships between them



Baseline Architecture of Scene Graph Generation

- 2 - stage framework
  - Object detection network extracts object regions and corresponding features
  - Message passing network with nodes initialized with these region features and edges accounting for the potential relations among them

- Procedure
  - Object detection → bounding box
  - Object label classification
  - Initialize a set of node features (state initialization)
  - Refine features via context encoding e.g. LSTM, GNNs
  - Cross entropy loss



Motivation

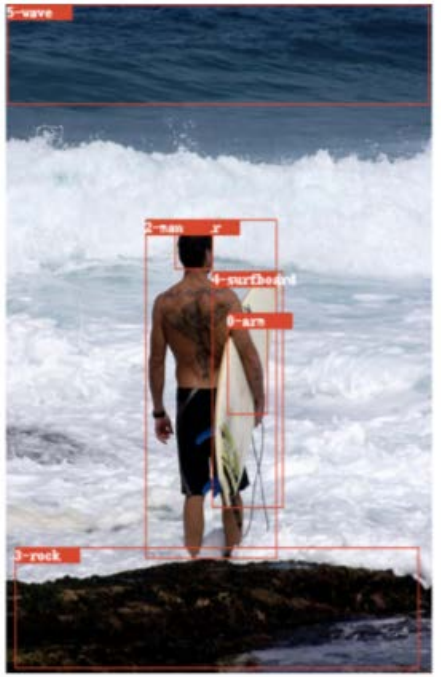
< Loss of existing scene graph generation model >

cross-entropy loss that treats objects and relationships in a scene graph as independent entities

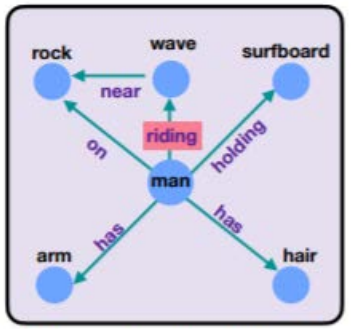
$$\log p(SG|I) = \sum_{i \in O} \log p(o_i|I) + \sum_{j \in R} \log p(r_j|I)$$

< Limitation >

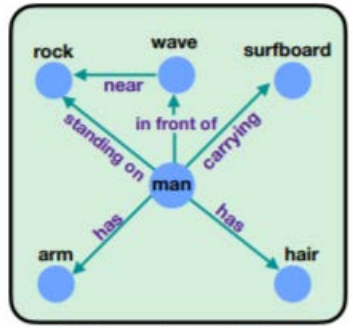
1) Ignores the structure of the scene graph output space (e.g., correlation or exclusion among object and relation label sets)
2) The imbalance in the number of training samples for the relations results in dominant relations being heavily favored, leading to biased relation prediction at test time



(a) Input Image



(b) Cross entropy based training



(c) Energy based training

Motivation

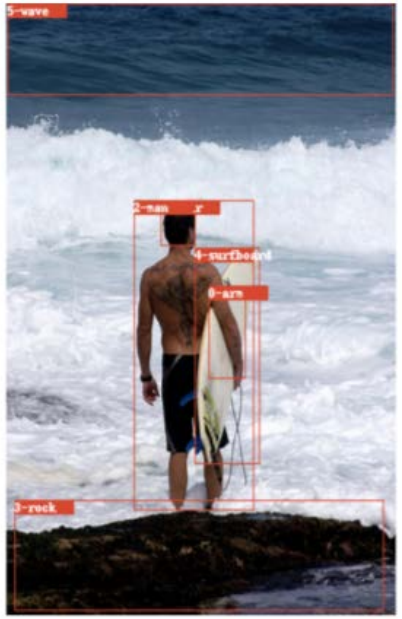
< Limitations >

1) During loss computation, the loss for each relation term is independent of the relations predicted in the rest of the scene graph

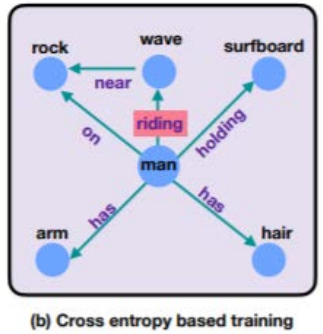
<man, riding, wave> ≈ <man, behind, wave> given <man, carrying, surfboard>

2) Due to the summation over individual relation terms, the model, in order to minimize the loss, is incentivized to predict relations which are more common in the training data

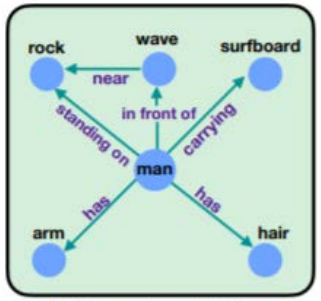
→ Suggested energy-based modeling as a solution



(a) Input Image



(b) Cross entropy based training



(c) Energy based training

Energy-Based Model (EBM)

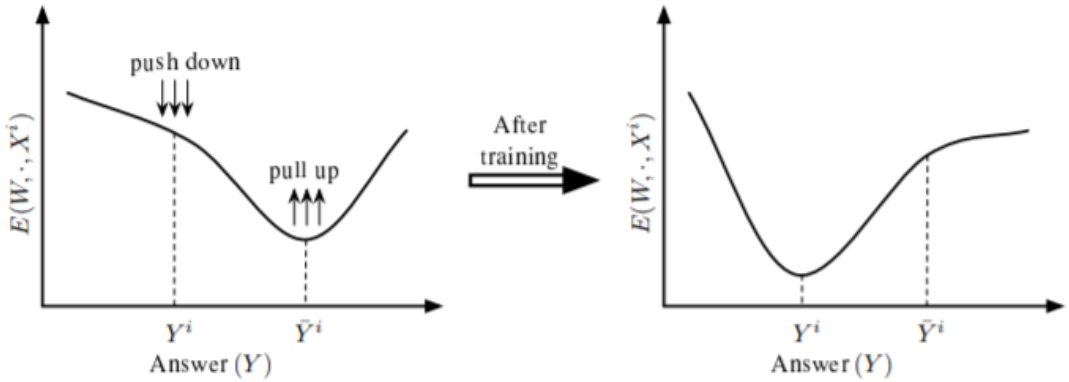
- Encode dependencies between variable by assigning a scalar energy value to an input configuration
- Boltzmann distribution

$$p_\theta(x,y) = \frac{exp\big(-E_\theta(x,y)\big)}{Z(\theta)} \quad where\ Z(\theta) = \int exp\big(-E_\theta(x,y)\big)$$

- Using MCMC that sample from the data distribution

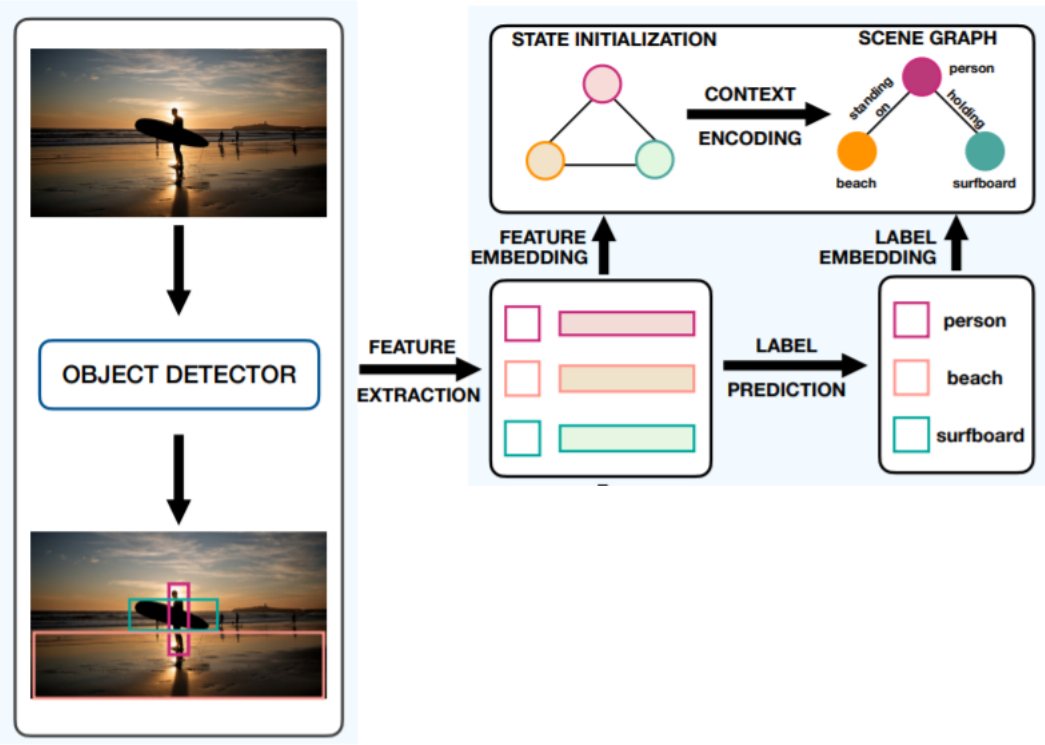
$$\nabla_\theta log p_\theta(x,y) = \mathbb{E}_{p_\theta(x\prime,y\prime)}[\nabla_\theta E_\theta(x,y)] - \nabla_\theta E_\theta(x,y)$$

- High energy states correspond to less stable states



Method - Architecture

- A simple implementation of the joint energy function: take an encoding of image and a scene graph and produce a scalar energy value
- Challenges: 1) fail to capture small regions, 2) SG is variable in length and high dimensional
- Two additional units: extracting an image graph G_I, energy computation unit

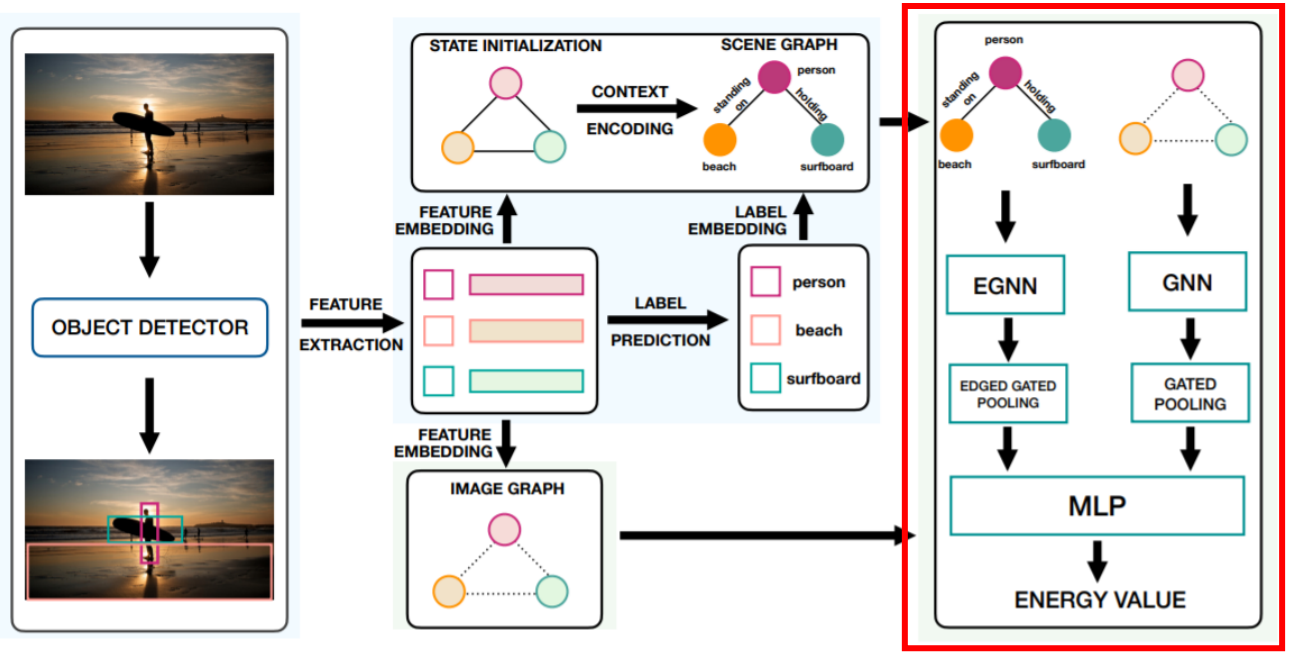


Method - Energy Model Architecture

- Given an image graph and a scene graph, the energy model refines the state representations using graph neural networks
- Edge Graph Neural network (EGNN) and Graph Neural Network (GNN)

$$E_\theta(G_I, G_{SG}) = MLP\left[f\left(EGNN(G_{SG})\right); g\left(GNN(G_I)\right)\right]$$

- Edge Graph Neural network (EGNN): variant of graph message passing algorithm
- Use gated pooling layers to generate vector representations of the two graphs



https://arxiv.org/pdf/1901.00596.pdf

Method – loss

- Find a scene graph configuration that minimizes the energy value → Stochastic Gradient Langevine Dynamics (SGLD)

$$\mathcal{L}_e = E_\theta(G_I^+, G_{SG}^+) - \min_{G_{SG} \in SG} E_\theta(G_I, G_{SG})$$

G_I^+, G_{SG}^+ are ground truth image graph and scene graph, respectively

- L2 regularization loss on the energy values

$$\mathcal{L}_r = E_\theta(G_I^+, G_{SG}^+)^2 + E_\theta(G_I, G_{SG})^2$$

- Total loss

$$\mathcal{L}_{total} = \lambda_e \mathcal{L}_e + \lambda_r \mathcal{L}_r + \lambda_t \mathcal{L}_t$$

$\mathcal{L}_t$ denotes task loss used by the underlying scene graph generation model

Result

Datasets: Visual Genome, GQA

Dataset	Model	Method	Predicate Classification			Scene Graph Classification			Scene Graph Detection		
			mR@20	mR@50	mR@100	mR@20	mR@50	mR@100	mR@20	mR@50	mR@100
Visual Genome	VCTree [22]	Cross Entropy	13.07	16.53	17.77	8.5	10.53	11.24	5.31	7.16	8.35
		EBM-Loss	**14.2**	**18.19**	**19.72**	**10.4**	**12.54**	**13.45**	**5.67**	**7.71**	**9.1**
	Motif [31]	Cross Entropy	12.45	15.71	16.8	6.95	8.85	9.05	5.07	6.91	8.12
		EBM-Loss	**14.17**	**18.02**	**19.53**	**8.18**	**10.22**	**10.98**	**5.66**	**7.72**	**9.27**
	IMP [26]	Cross Entropy	8.85	10.97	11.77	5.4	6.4	6.74	2.2	3.29	4.14
		EBM-Loss	**9.43**	**11.83**	**12.77**	**5.66**	**6.81**	**7.17**	**2.78**	**4.23**	**5.44**
	VCTree-TDE [21]	Cross Entropy	16.3	22.85	26.26	11.85	15.81	17.99	6.59	8.99	10.78
		EBM-Loss	**19.87**	**26.66**	**29.97**	**13.86**	**18.2**	**20.45**	**7.1**	**9.69**	**11.6**
GQA	Transformer [23]	Cross Entropy	1.17	2.48	3.69	.54	.97	1.29	-	-	-
		EBM-Loss	**1.28**	**2.94**	**4.71**	**.68**	**1.32**	**1.77**	-	-	-
	Motif [31]	Cross Entropy	.85	1.8	2.75	.42	.81	1.18	-	-	-
		EBM-Loss	**.94**	**2.1**	**3.19**	**.57**	**.9**	**1.26**	-	-	-
	IMP [26]	Cross Entropy	.5	.94	1.32	.28	.5	.65	-	-	-
		EBM-Loss	**.57**	**1.07**	**1.5**	**.34**	**.58**	**.76**	-	-	-

Table 1. **Quantitative Results.** We compare the proposed energy-based loss formulation against traditional cross-entropy loss using various state-of-the-art models. We report the mean Recall@K [22] under all three experimental setting.

Result

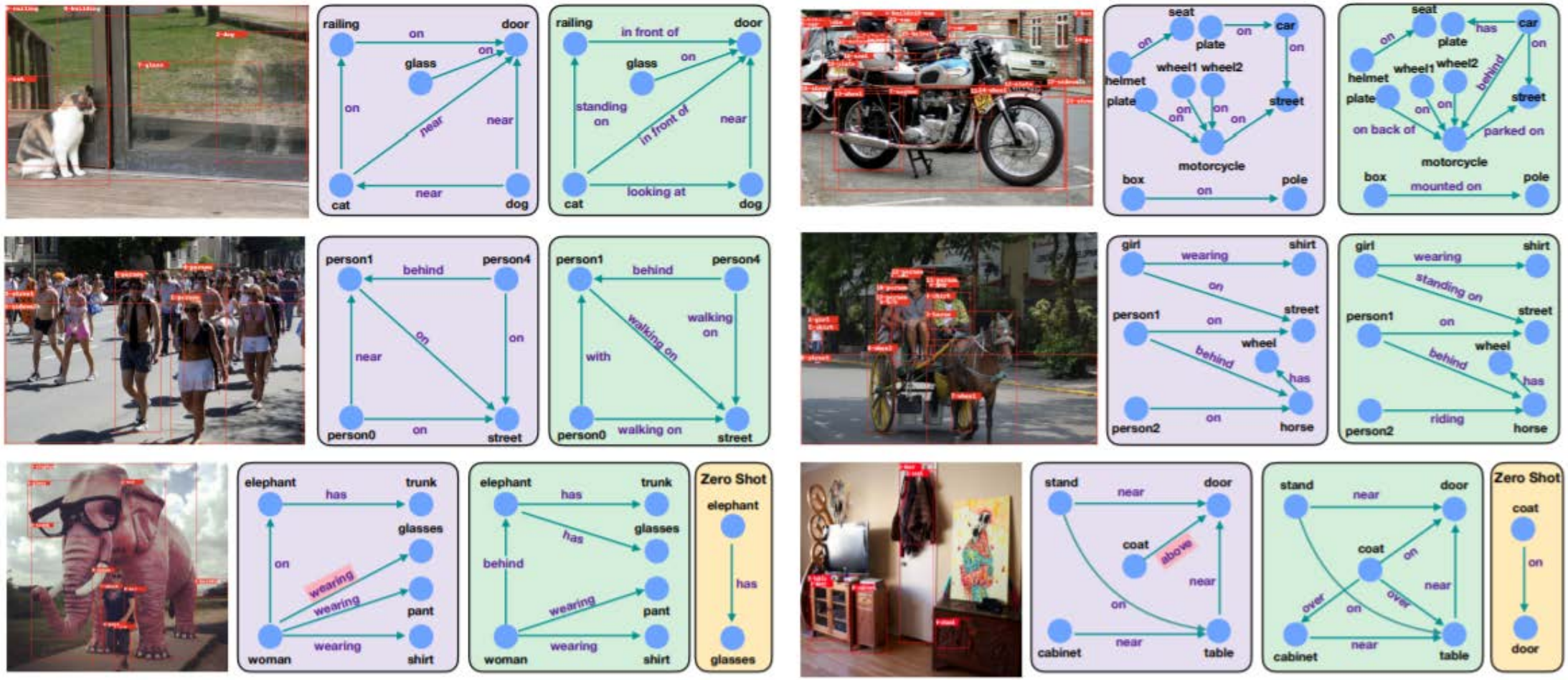
Dataset	Model	Method	PredCls zsR@20/50	SGCls zsR@20/50	SGDet zsR@20/50
VG	VCTree	CE	1.43/4	.39/1.2	.19/.46
		EB	**2.25/5.36**	**.87/1.87**	**.21/.54**
	Motif	CE	1.28/3.56	.39/.83	0/.04
		EB	**2.07/4.87**	**.52/1.25**	**.11/.23**
	IMP	CE	12.17/17.66	2.09/3.3	.14/.39
		EB	**12.6/18.6**	**2.29/3.7**	**.16/.43**
	VCTree-TDE	CE	8.98/14.52	3.16/4.97	1.47/2.3
		EB	**9.58/15.14**	**4.18/6.38**	**1.62/2.68**
GQA	Transformer	CE	19.55/33.33	.94/1.83	-
		EB	**20.11/34.33**	**1.2/2.05**	-
	Motif	CE	17.74/30.61	1.27/2.16	-
		EB	**19.47/33.45**	**1.49/2.48**	-
	IMP	CE	15.58/27.6	1.02/1.88	-
		EB	**16.65/27.77**	**1.1/1.98**	-

Table 2. **Zero-shot Recall.** The zero shot recall performance comparison of model trained using cross-entopy (CE) and energy-based loss (EB) on the Visual Genome (VG) and GQA dataset.

	Few-Shot Recall@20				
$k_1 - k_2$ shot	1-5	6-10	11-15	16-20	20-25
C.E.	16.9	24.41	27.73	31.52	32.31
E.B.M.	**18.55**	**25.22**	**28.1**	**32.05**	**32.57**

Table 3. **Few-shot Recall@20.** Table compares the few short recall performance of a VCTree [22] model trained using cross-entropy and energy-based loss.

Result – qualitative analysis



purple: GCTree using CE loss, green: proposed energy-based model, yellow: zero-shot triplet

THANK YOU

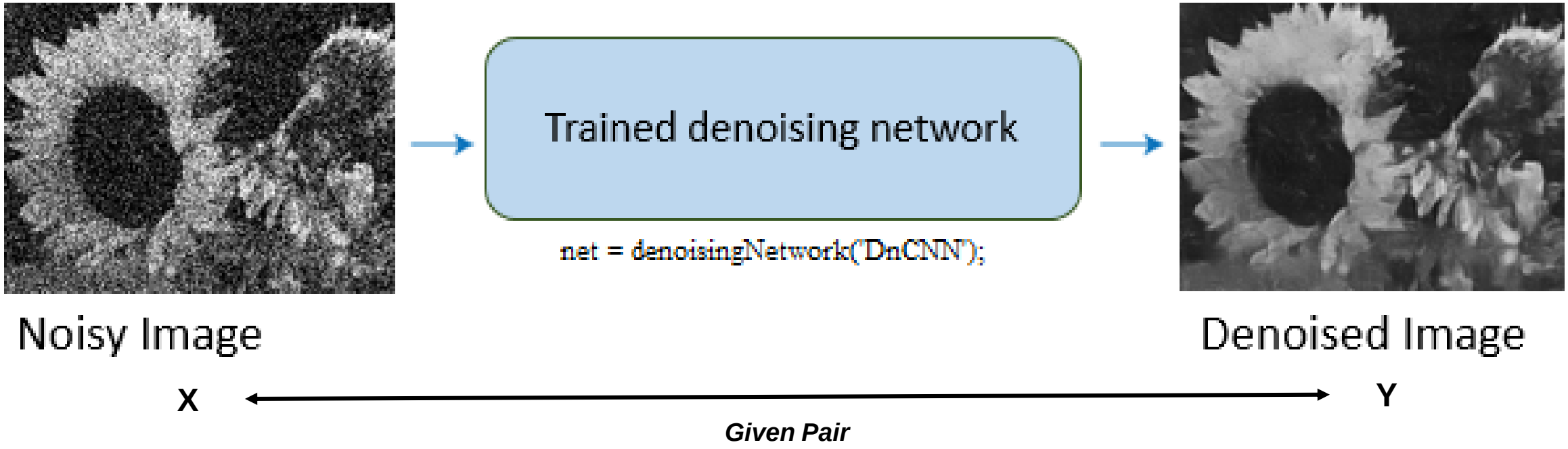
Self-Supervised Image Prior Learning with GMM from a Single Noisy Image

ICCV 2021

이근택

Introduction - Denoising

- Noisy image로부터 Clean image를 출력하는 Task
- 일반적으로 Noisy Image <-> Clean Image Pair를 통해 Supervised 방법으로 학습
- Real World에서는 이러한 Noisy-Clean Pair를 구하기 힘듬(Not Practical)



Introduction - Denoising

- Self-supervised Denoising – Noisy Image로부터 Image Prior를 학습
- 학습에서 GT Clean Image는 주어지지 않음
- Noisy Image만 주어져도 Denoising을 수행 할 수 있음

Noisy Image

Given Only X

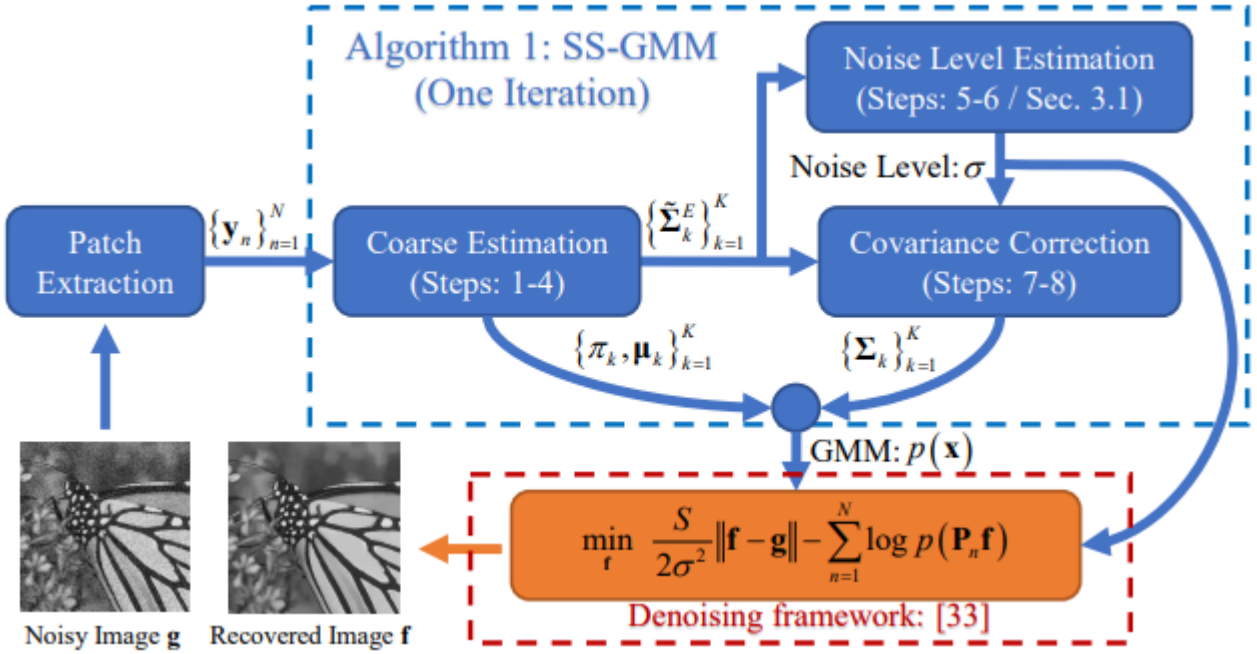
Trained denoising network

```
net = denoisingNetwork('DnCNN');
```

Denoised Image

Not Given Y

Overall Schematic



- Using GMM to represent distribution of image patch
- Learn GMM parameters from only a single noisy image
- Using constrained-added Expectation-Maximization (EM-GMM) to estimate parameters of GMM
- Estimate image prior with GMM, using this prior to denoise image
- Analyze learned parameters with influence of noise

GMM with Image Patch

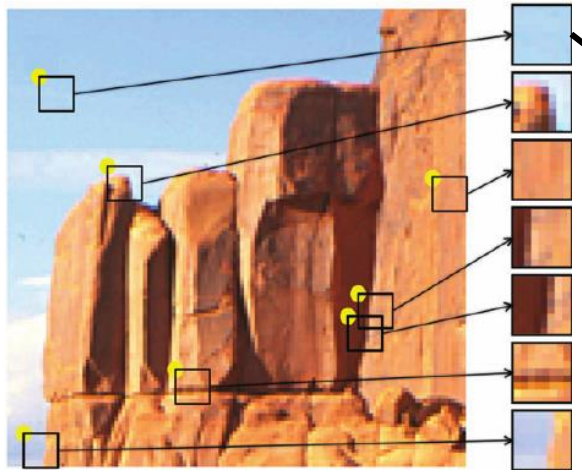
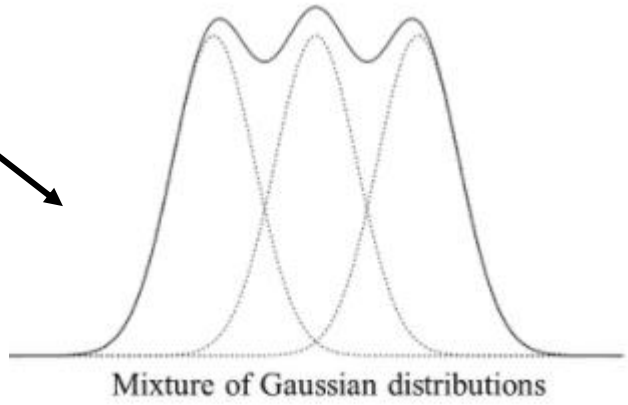


Image Patch



$\{\boldsymbol{\pi_k}, \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k} + \boldsymbol{\sigma^2} \cdot \mathbf{I}\}$

Clean Image

$\Sigma_k = D_k \Lambda_k D_k^T$

$\Lambda_k = \{\lambda_{ks}\}_{s=1}^{S}$

Noisy Image with level σ^2

$\tilde{\Sigma}_k = \Sigma_k + \sigma^2 \cdot \mathrm{I}$

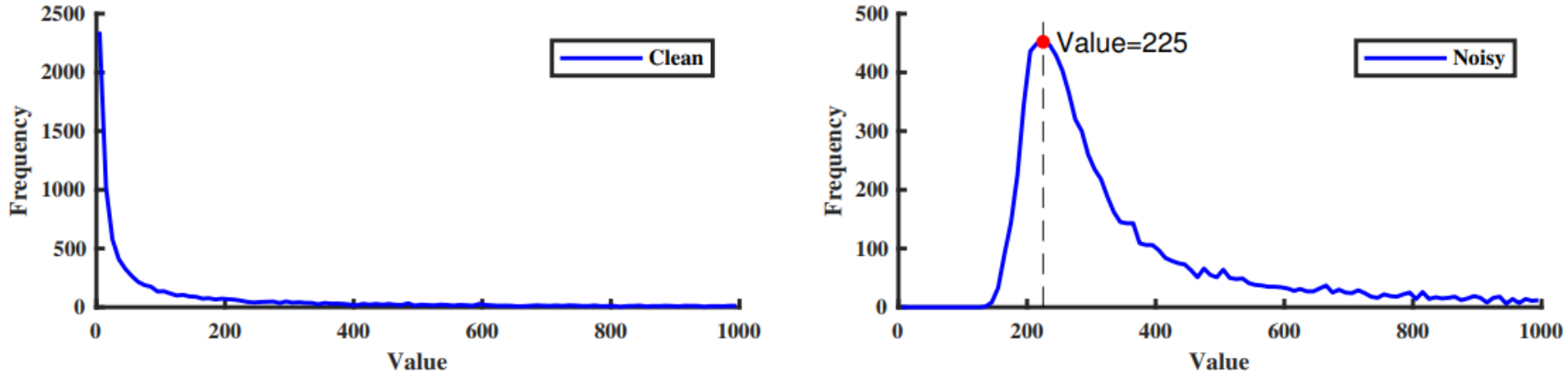
$\tilde{\Sigma}_k = \Sigma_k + \sigma^2 \cdot \mathrm{I} = D_k \tilde{\Lambda}_k D_k^T$

$\tilde{\Lambda}_k = \{\tilde{\lambda}_{ks}\}_{s=1}^{S}$

$\boldsymbol{\tilde{\lambda}_{ks} = \lambda_{ks} + \sigma^2}$

- GMM can fit any distribution
- Assume image patch belongs to a GMM with some parameters
- We can decompose covariance matrices of clean image and noisy image
- Eigenvector of clean image and noisy image is same
- How to decouple λ_{ks} and σ^2?

GMM with Image Patch



(a) Histogram of λ^E (b) Histogram of $\tilde{\lambda}^E$

Figure 3. Histograms of covariance eigenvalues learned by EM-GMM from (a) clean images and (b) images added with Gaussian noise of the noise level $\sigma = 15$.

- Covariance eigenvalue learned from clean image with EM algorithm shows λ_{ks} accumulate around zero -> Hold sparsity
- However, eigenvalue from noisy image gather around σ^2 where peak frequency occurs
- We only can estimate empirically with training samples
- Training samples should be large enough

GMM with Image Patch

Covariance matrix of well trained GMM

$$\Sigma_k^E \approx \frac{1}{N_k} \sum_{n \in S_k} (\mathrm{X}_n - \mu_k)(\mathrm{X}_n - \mu_k)^T$$

Eigenvalue of Σ_k^E, Approx Gaussian

$$p(\lambda_{ks}^E) \approx \mathcal{N}\left(\lambda_{ks}^E; \lambda_{ks}, \frac{2\lambda_{ks}^2}{N_k}\right)$$

Randomly picked eigenvalue from clean image

$$p(\lambda^E) \approx \frac{1}{S \cdot N} \sum_{k=1}^{K} \sum_{s=1}^{S} \mathcal{N}\left(\lambda_{ks}^E; \lambda_{ks}, \frac{2\lambda_{ks}^2}{N_k}\right)$$

Randomly picked eigenvalue from noisy image

$$p(\tilde{\lambda}^E) \approx \frac{1}{S \cdot N} \sum_{k=1}^{K} \sum_{s=1}^{S} \mathcal{N}\left(\tilde{\lambda}_{ks}^E; \tilde{\lambda}_{ks}, \frac{2\tilde{\lambda}_{ks}^2}{N_k}\right)$$

$\boldsymbol{N}$: # of training samples

$\boldsymbol{S_k}$: Set containing all of the patches belong to k-th Gaussian component

$\boldsymbol{N_k}$: # of patches in S_k

→ $\boldsymbol{Most\ of\ \lambda_{ks}\ close\ to\ 0, \tilde{\lambda}_{ks}\ approximate\ to\ \sigma^2}$

$$\tilde{\lambda}_{ks} = \lambda_{ks} + \sigma^2 \approx \sigma^2$$

$$\approx 0$$

- Following equation of GMM with EM, we can empirically estimate covariance matrix
- Each randomly picked eigenvalue has approximately Gaussian distribution
- As shown in histogram, eigenvalue of clean image close to zero
- Can show eigenvalue of noisy image will be approximate to it's noise level σ^2

Self-Supervised Learning Method

Classic EM-GMM likelihood function

$$\max_{\pi_k,\mu_k,\tilde{\Sigma}_k} \frac{1}{N}\sum_{n=1}^{N} \ln\left(\sum_{k=1}^{K} \pi_k \cdot \mathcal{N}\left(y_n;\mu_k,\tilde{\Sigma}_k\right)\right),$$

$$s.t.\ 0 \le \pi_k \le 1, \sum_{k=1}^{K} \pi_k = 1$$

➡ ***Vulnerable to noise***

Add constraint

Self-Supervised EM-GMM likelihood function

$$\max_{\theta} \frac{1}{N}\sum_{n=1}^{N} \ln\left(\sum_{k=1}^{K} \pi_k \cdot \mathcal{N}\left(y_n;\mu_k,\tilde{\Sigma}_k\right)\right),$$

$$s.t.\ 0 \le \pi_k \le 1, \sum_{k=1}^{K} \pi_k = 1\,,$$

$$\tilde{\Sigma}_k = D_k \cdot \tilde{\Lambda}_k \cdot D_k^T, \qquad D_k^T \cdot D_k = \mathrm{I},$$

$$\tilde{\Lambda}_k = \Lambda_k + \sigma^2 \cdot \mathrm{I}, \qquad \lambda_{ks} \ge 0,$$

$$\sum_{k=1}^{K}\sum_{s=1}^{S} \|\lambda_{ks}\|_0 \le L$$

- As shown in histogram, covariance estimated by EM-GMM is vulnerable to noise
- Define constraint term with relationship between noisy/clean image
- Add constraint that eigenvector maintain sparsity
- Optimize **Self-Supervied EM-GMM** with conventional EM framework

Self-Supervised Learning Method

E-step

- *Determine* γ_{nk}

$$\gamma_{nk} = \frac{\pi_k \cdot \mathcal{N}(y_n; \mu_k, \tilde{\Sigma}_k)}{\sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(y_n; \mu_k, \tilde{\Sigma}_k)}$$

M-step

- *Maximize below function*

$$Q(\theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left(\ln \pi_k + \ln \mathcal{N}(y_n; \mu_k, \tilde{\Sigma}_k) \right)$$

GMM parameter $\theta = \{\pi_k, \mu_k, D_k, \Lambda_k\}$

Sub problems in M-step

- *Sub-1: Optimizing* π_k, μ_k

- *Sub-2: Optimizing* D_k

$$\min_{D_k} Tr\left(\tilde{\Lambda}_k^{-1} D_k^T \tilde{\Sigma}_k^E D_k\right)$$
$$s.t. \;\; D_k^T \cdot D_k = \mathrm{I}$$
$$\rightarrow D_k = \tilde{D}_k^E$$

- *Sub-3: Optimizing* λ_k, σ^2

$$\lambda_{ks} = \begin{cases} 0 & , \tilde{\lambda}_{ks}^E < \tilde{\lambda}_L^E \\ \tilde{\lambda}_{ks}^E - \sigma^2 & , \tilde{\lambda}_{ks}^E \geq \tilde{\lambda}_L^E \end{cases}$$

$$\sigma^2 = \frac{\sum_{k=1}^{K} \sum_{s=1}^{S} \left[\tilde{\lambda}_{ks}^E < \tilde{\lambda}_L^E\right] \cdot \tilde{\lambda}_L^E}{\sum_{k=1}^{K} \sum_{s=1}^{S} \left[\tilde{\lambda}_{ks}^E < \tilde{\lambda}_L^E\right]}$$

Algorithm 1: Self-Supervised GMM (SS-GMM)

Input: Noisy image patches $\{\mathbf{y}_n\}_{n=1}^N$
Output: Noise level σ^2,
GMM parameters $\{\pi_k, \boldsymbol{\mu}_k, \mathbf{D}_k, \boldsymbol{\Lambda}_k\}_{k=1}^K$
Initialize parameters with EM-GMM [2, 3];
while *not converge* **do**
1). Calculate probability γ_{nk} as Eq. (11); — ***E-step***
2). Calculate mixing coefficient π_k as Eq. (10a);
3). Calculate mean vectors $\boldsymbol{\mu}_k$ as Eq. (10b);
4). Calculate covariance $\tilde{\boldsymbol{\Sigma}}_k^E$ as Eq. (10c);
5). Do eigenvalue decomposition for $\tilde{\boldsymbol{\Sigma}}_k^E$; — ***M-step***
6). Estimate noise level as Sec. 3.1 introduces;
7). Determine parameter L by Eq. (19);
8). Calculate eigenvalues λ_{ks} as Eq. (18);
end

$$[A] = \begin{cases} 1, & if\ A\ is\ true; \\ 0, & otherwise \end{cases}$$

Iverson Bracket

- Iterate E-step and M-step until converge
- Initialize parameters with EM-GMM
- We can estimate noise level and GMM parameters by EM algorithm
- Many parameters optimized in M-step, separate M-step to several sub-problems

Experimental Results

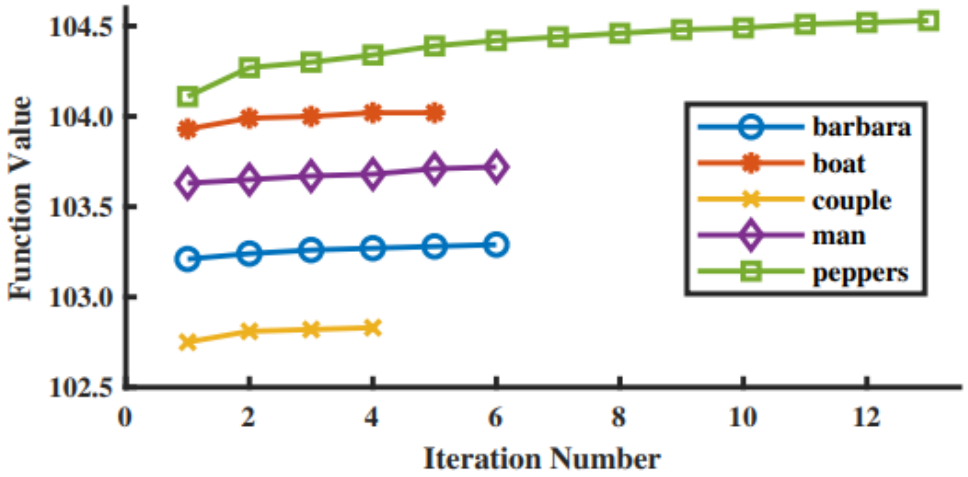


Figure 4. Objective function values versus iteration numbers.

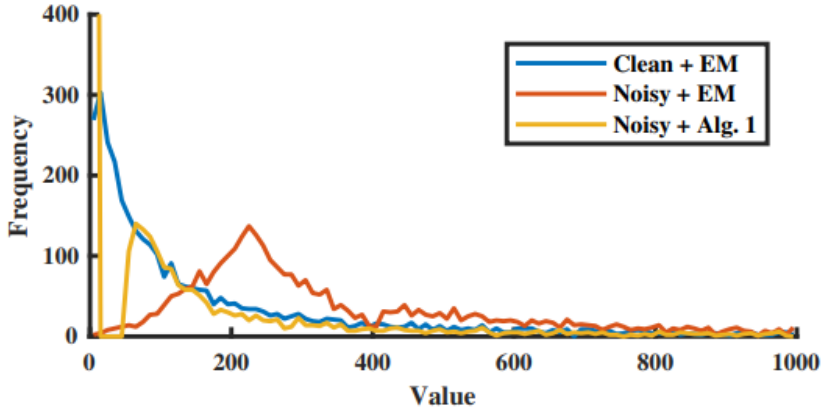


Figure 5. Histograms of eigenvalues learned by Alg. 1 and those learned from the clean/noisy 'couple' with the EM-GMM algorithm. The noise level is $\sigma = 15$.

- Objective function value converges for any images with stability
- Conventional EM algorithm peaks on noise value
- SS EM-GMM converge fast and hold sparsity
- Figure shows SS EM-GMM can decouple eigenvector from noise level

Experimental Results

Table 1. Image Denoising Performances on Set12 and BSD68. The best results are highlighted in bold. The results marked with '*' are quoted from [22]. Comparison methods include BM3D [8], EPLL [33], PGPD [30], NL-Bayes [16], N2V [15], DIP [28], S2S [22] and our proposed SS-GMM.

Dataset	σ	BM3D	EPLL	PGPD	NL-Bayes	N2V	DIP	S2S	SS-GMM
Set12	15	32.12	31.83	32.13	31.98	30.73	30.90	31.83	**32.18**
	25	29.73	29.38	29.69	29.61	28.86	28.89	**29.75**	29.68
	50	26.49	26.09	**26.53**	26.35	26.00	25.52	26.38	26.38
BSD68	15	31.08	31.22	31.13	31.14	29.25	29.70	30.26	**31.26**
	25	28.56	28.72	28.62	28.69	27.69	28.00	28.70*	**28.73**
	50	25.62	25.72	25.75	25.66	25.44	25.08	**25.92***	25.70

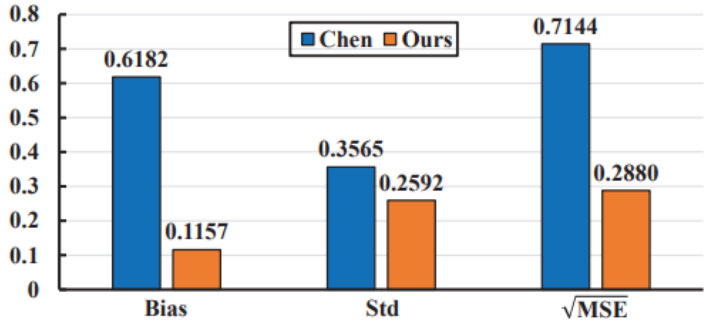


Figure 7. Noise level estimation results on Set12. The smaller these measurements are, the better an estimator is.

- SS-GMM can estimate noise level more accurately
- SS-GMM achieve robust and better performance on diverse noise level and image

Visual Results

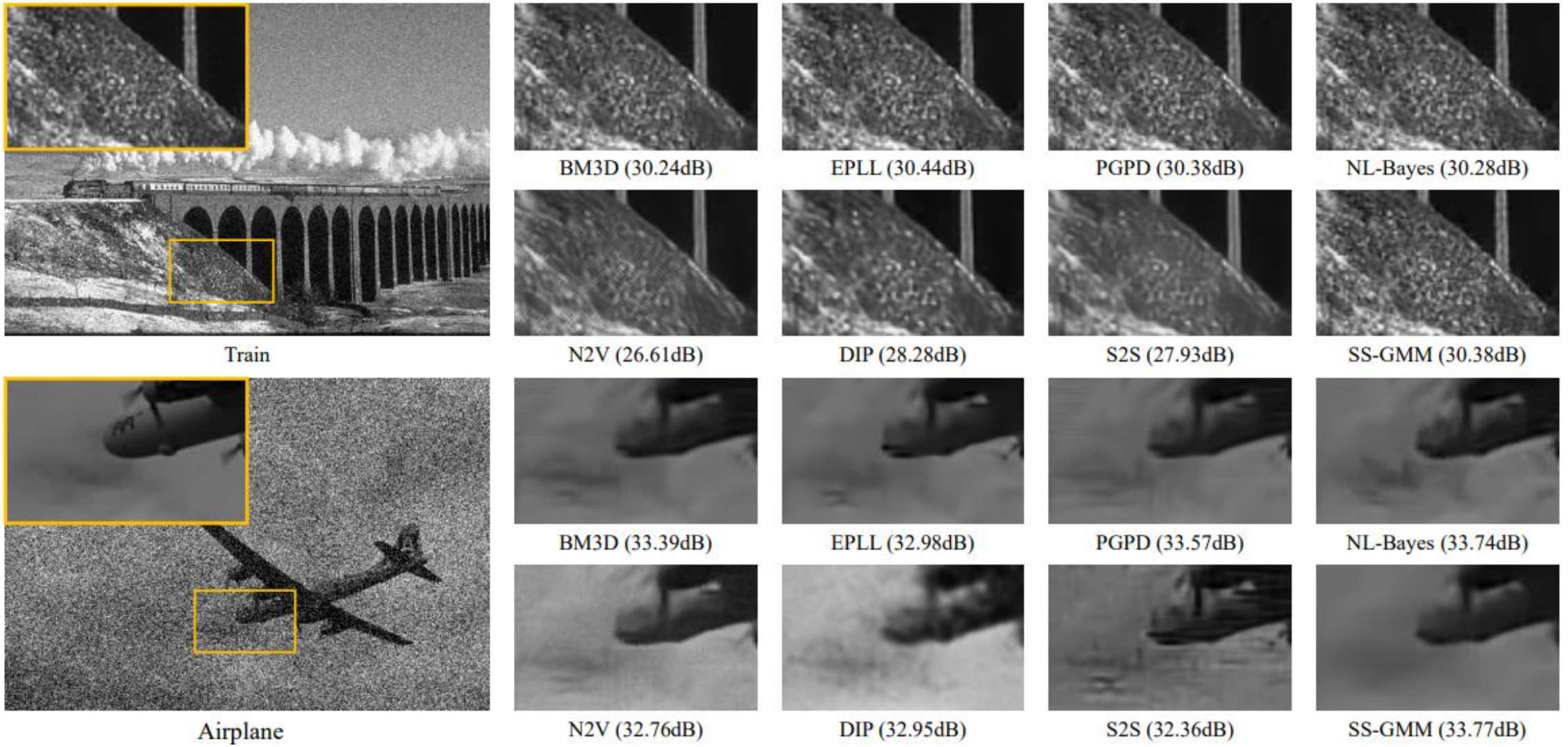


Figure 8. Visual results of comparison algorithms on regions of the image 'Train' ($\sigma = 15$) and regions of the image 'Airplane' ($\sigma = 50$). The inserts shown in the first column are clean GT regions. The whole images for these regions are provided in the supplementary materials.

Thank You!

What Should Not Be Contrastive In Contrastive Learning

Tete Xiao[1], Xiaolong Wang[2], Alexei A. Efros[1], Trevor Darrell[1]
UC Berkeley[1], UC San Diego[2]
Published at ICLR 2021

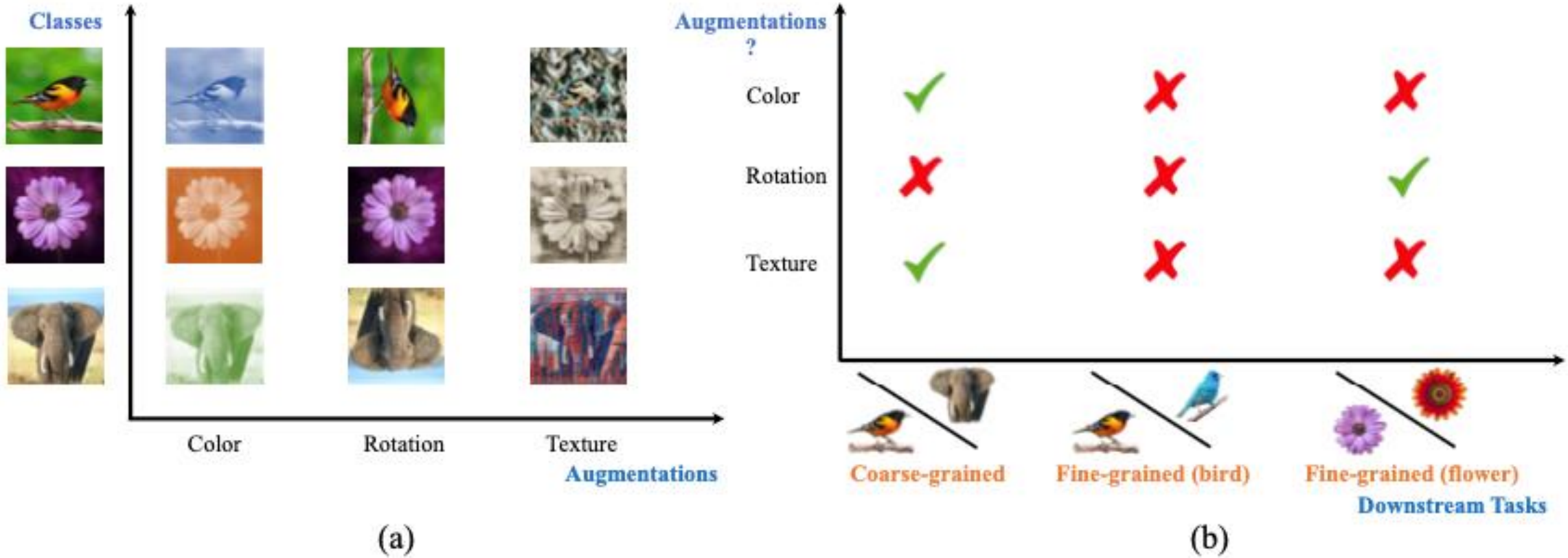
2021학년도 2학기 패턴인식 논문 발표

협동과정 인공지능 전공 석사과정

이상준 (2021-24432)

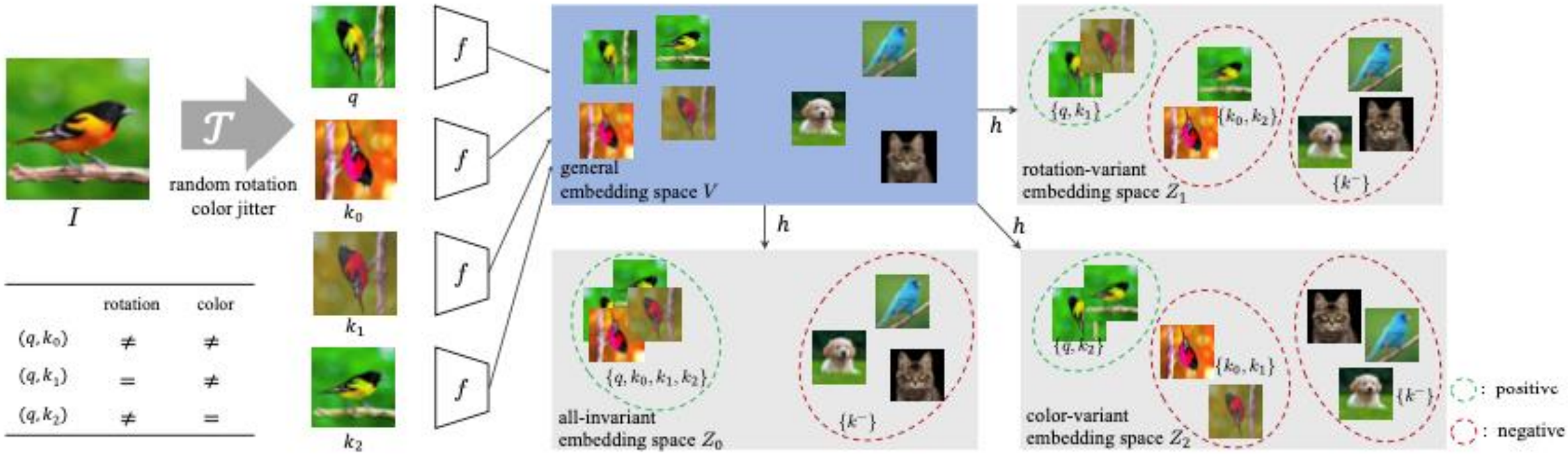
Introduction

Double-edged sword



LooC: Leave-One-Out Contrastive Learning

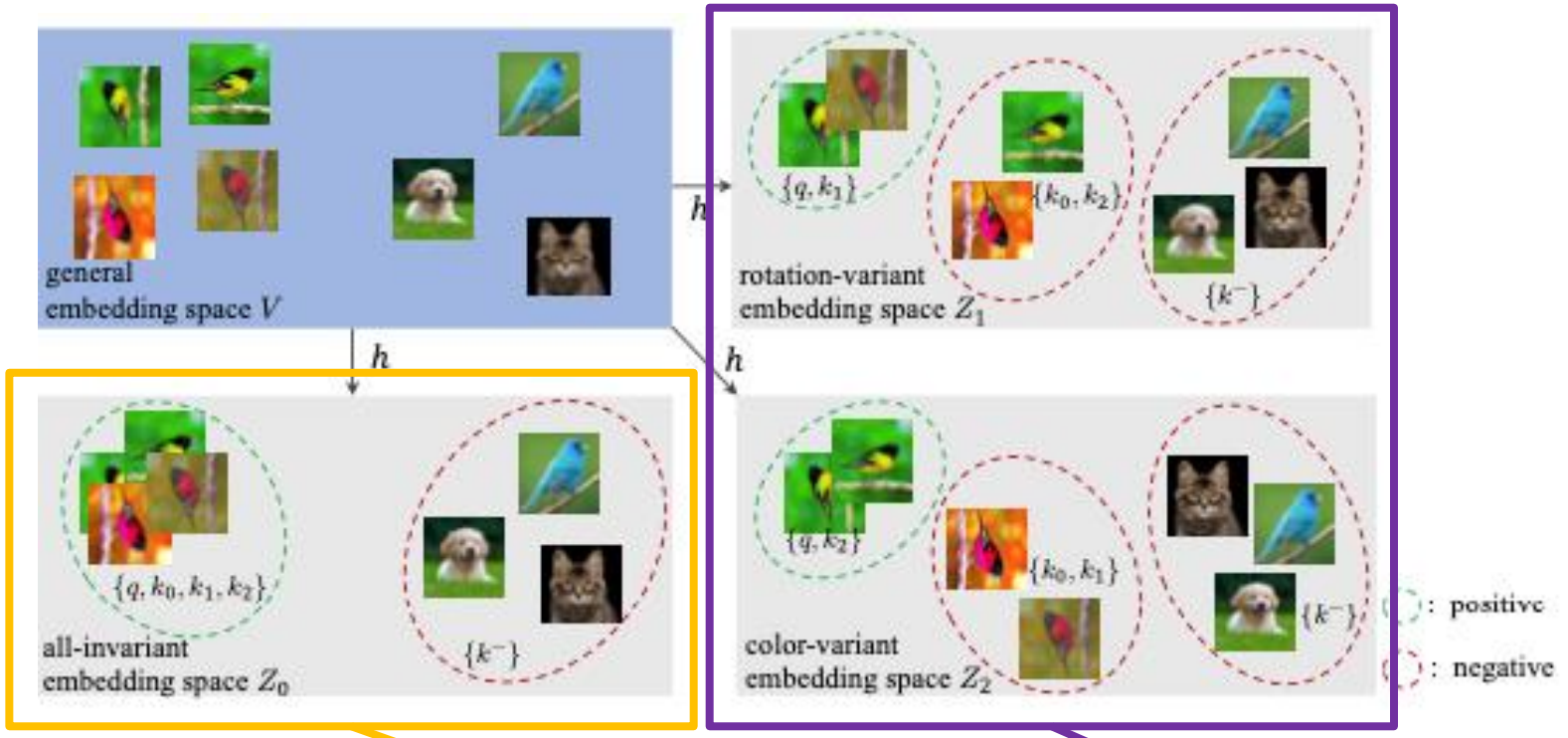
Model Architecture



	rotation	color
(q, k_0)	$\neq$	$\neq$
(q, k_1)	$=$	$\neq$
(q, k_2)	$\neq$	$=$

LooC: Leave-One-Out Contrastive Learning

Overall training objective



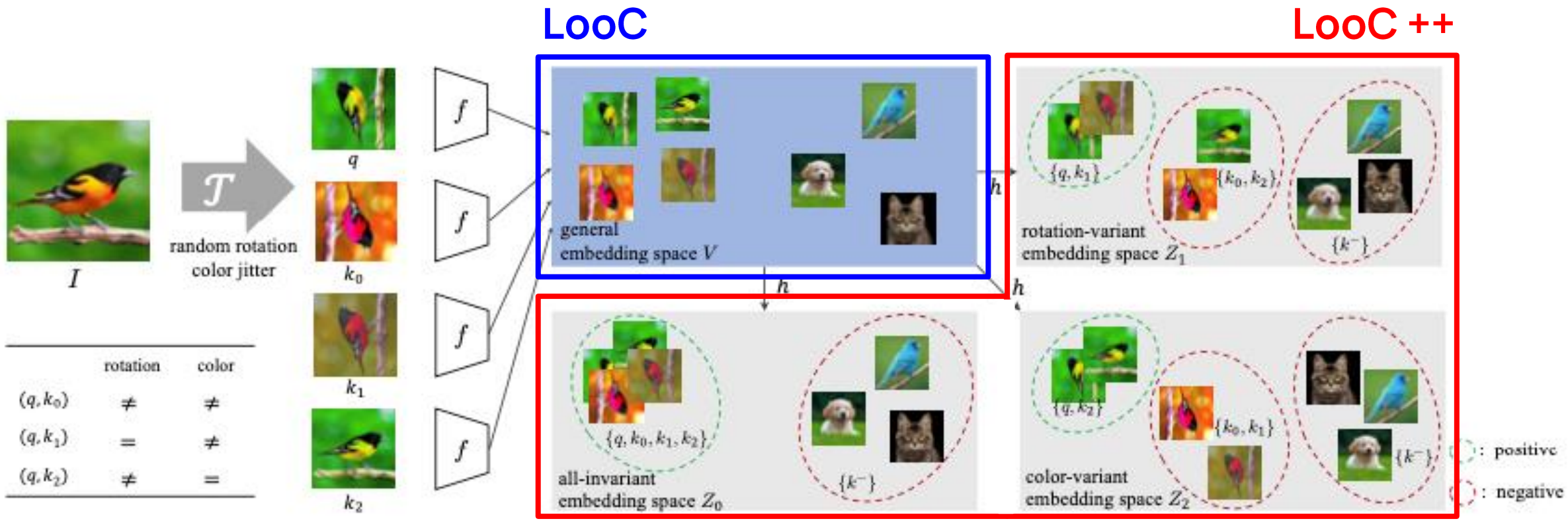
- In Z_0,
  - the Positive pair : z_0^q & $z_0^{k_0^+}$
  - Negative pairs: $\{z_0^{k_0^-}\}$
- In Z_i ($\forall i \in \{0,1,\cdots,n\}$),
  - the Positive pair : z_i^q & $z_i^{k_i^+}$
  - Negative pairs: $\{z_i^{k_i^-}\}$, $\{z_i^{k_j^+}\}$

$$\mathcal{L}_q = -\frac{1}{n+1}\left(\log \frac{E_{0,0}^+}{E_{0,0}^+ + \sum_{k-} E_{0,0}^-} + \sum_{i=1}^{n} \log \frac{E_{i,i}^+}{\sum_{j=0}^{n} E_{i,j}^+ + \sum_{k-} E_{i,i}^-}\right)$$

$$E_{i,j}^{\{+,-\}} = \exp\left(\boldsymbol{z}_i^q \cdot \boldsymbol{z}_i^{k_j^{\{+,-\}}} / \tau\right)$$

LooC: Leave-One-Out Contrastive Learning

Learnt representation for downstream tasks



	rotation	color
(q, k_0)	$\neq$	$\neq$
(q, k_1)	$=$	$\neq$
(q, k_2)	$\neq$	$=$

Experiments

model	Rotation Acc.	IN-100 top-1	IN-100 top-5
Supervised	72.3	83.7	95.7
MoCo	61.1	81.0	95.2
MoCo + Rotation	43.3	79.4	94.1
MoCo + Rotation (same for q and k)	45.5	78.1	94.3
LooC + Rotation [ours]	65.2	80.2	95.5

model	Augmentation Color	Augmentation Rotation	iNat-1k top-1	iNat-1k top-5	CUB-200 top-1	CUB-200 top-5	Flowers-102 5-shot	Flowers-102 10-shot	IN-100 top-1	IN-100 top-5
MoCo	✓		36.2	62.0	36.7	64.7	67.9 (± 0.5)	77.3 (± 0.1)	81.0	95.2
LooC	✓		41.2	67.0	40.1	69.7	68.2 (± 0.6)	77.6 (± 0.1)	81.1	95.3
		✓	40.0	65.4	38.8	67.0	70.1 (± 0.4)	79.3 (± 0.1)	80.2	95.5
	✓	✓	44.0	69.3	39.6	69.2	70.9 (± 0.3)	80.8 (± 0.2)	79.2	94.7
LooC++	✓	✓	46.1	71.5	39.3	69.3	68.1 (± 0.4)	78.8 (± 0.2)	81.2	95.2

model	Aug. Rot.	Aug. Tex.	ON-13 top-1	ON-13 top-5	IN-C-100 (top-1) Noise	Blur	Weather	Digital	All	$d \geq 3$	IN-100 top-1	IN-100 top-5
Supervised			30.9	54.8	28.4	47.1	44.9	58.5	47.2	36.5	83.7	95.7
MoCo			29.2	54.2	37.9	38.5	47.7	60.1	48.2	37.2	81.0	95.2
LooC	✓		34.2	59.6	31.3	33.1	42.4	54.9	42.7	31.8	80.2	95.5
		✓	30.1	54.1	42.4	39.6	54.0	61.9	51.3	41.9	81.0	94.7
	✓	✓	33.3	59.2	37.0	35.2	50.2	56.9	46.5	37.2	79.4	94.3
LooC++	✓	✓	32.6	57.3	38.3	37.6	52.0	60.0	48.8	38.9	82.1	95.1

Thank you

Momentum Contrast for Unsupervised Visual Representation Learning

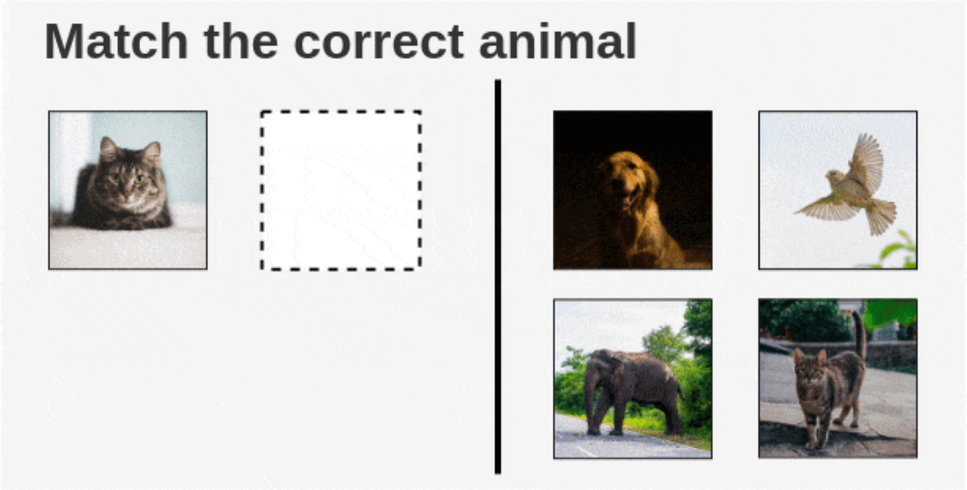
Pattern Recognition Assignment Paper Review

이휘준

What is Contrastive Learning?

비슷한 아이템들끼리는 '비슷하다!' → contrastive loss 적다

비슷하지 않은 아이템들끼리는 '비슷하지 않다!' → contrastive loss 크다.



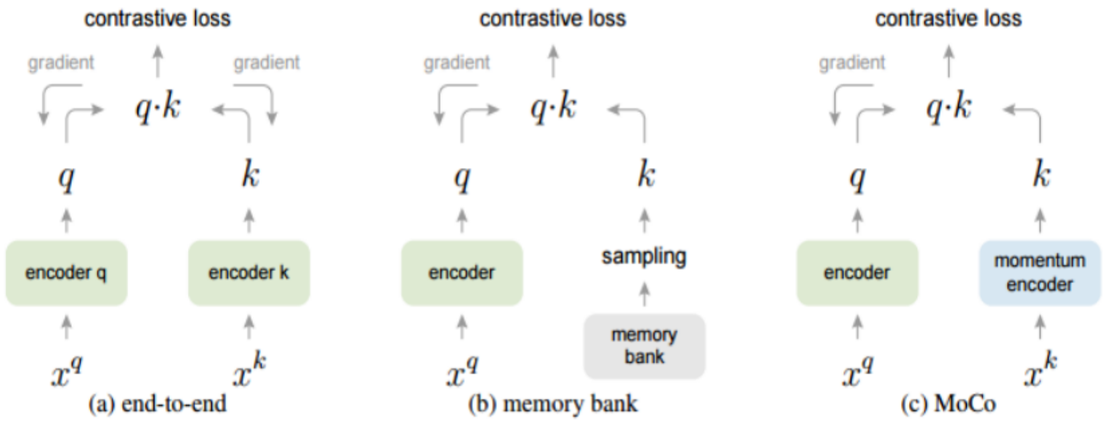


Figure 2. **Conceptual comparison of three contrastive loss mechanisms** (empirical comparisons are in Figure 3 and Table 3). Here we illustrate one pair of query and key. The three mechanisms differ in how the keys are maintained and how the key encoder is updated. **(a)**: The encoders for computing the query and key representations are updated ***end-to-end*** by back-propagation (the two encoders can be different). **(b)**: The key representations are sampled from a ***memory bank*** [61]. **(c)**: ***MoCo*** encodes the new keys on-the-fly by a momentum-updated encoder, and maintains a queue (not illustrated in this figure) of keys.

기존의 contrastive loss mechanism 방식: end-to-end (a) 그리고 memory bank (b)

* end-to-end 방식: negative sample이 너무 많이 필요하고 (Large batch) negative sample의 encoder는 query encoder와 일관성이 있어야함.(Inconsistency)

* memory bank 방식: 많은 양의 negative sample을 다룰 수 있지만 update된 negative sample이 encoder에 반영되지 않음.

→ 이와 같은 단점을 극복한 모델이 MoCo

Method

1. 이미지가 data augmentation (random crop, color jittering, flip, grayscale)에 의해 x_{query} x_{key} 로 나눠지게 된다.
2. 각각이 encoder(ResNet), momentum encoder를 거쳐 feature를 산출.
3. 같은 이미지에서 온 x_{query}, x_{key} 의 loss는 작고 다른 pair에 대해서는 크다 (다른 x_{key}는 미리 만들어 놓은 queue 형태의 dictionary에서 가져옴.)

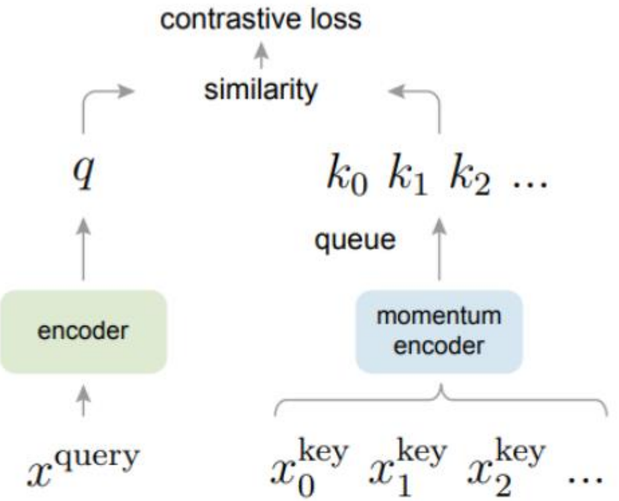


Figure 1. Momentum Contrast (MoCo) trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys $\{k_0, k_1, k_2, ...\}$ are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the current mini-batch enqueued and the oldest mini-batch dequeued, decoupling it from the mini-batch size. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder. This method enables a large and consistent dictionary for learning visual representations.

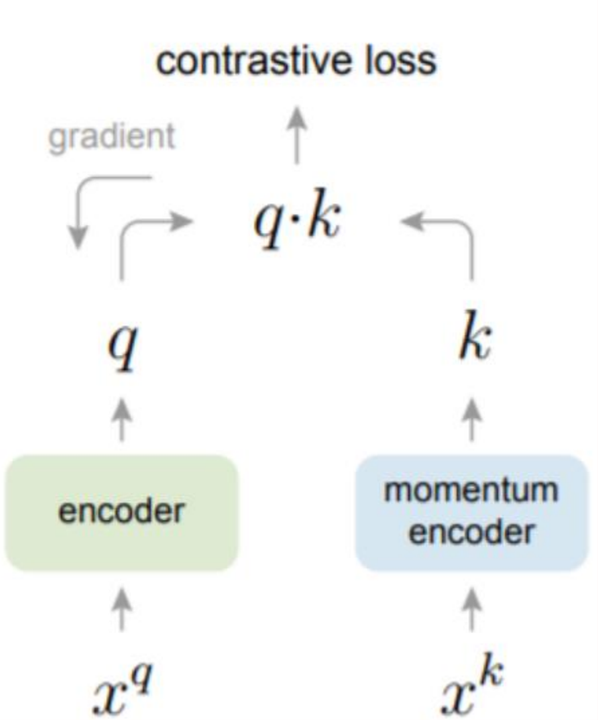
Method

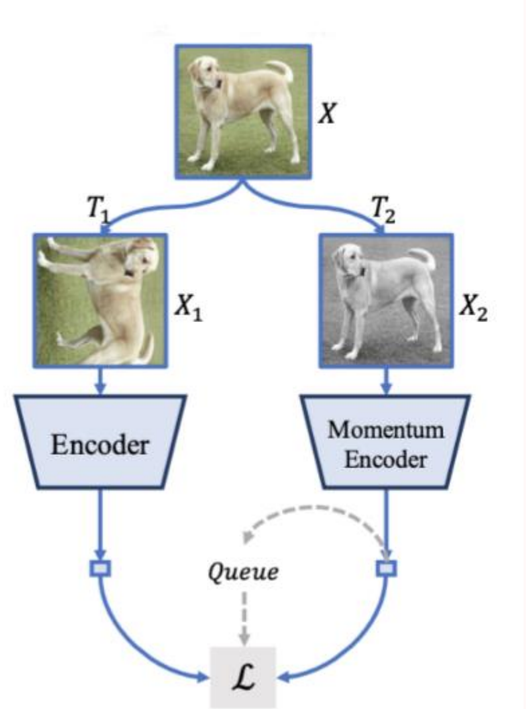
4. InfoNCE loss backpropagated into the encoder.

5. momentum encoder는 backpropagation 하지 않고, weight에 따라 update를 부분적으로 해준다.

→ *queue 구조를 활용한 dictionary 덕분에 충분한 양의 negative smaple로 학습 가능하고 dynamic dictionary로 inconsistency의 문제를 해결!*

→*크고 일관성 있는 dictionary로 단점을 극복!*





InfoNCE:

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

$$\theta_{\mathrm{k}} \leftarrow m\theta_{\mathrm{k}} + (1-m)\theta_{\mathrm{q}}.$$

Decision Transformer: Reinforcement Learning via Sequence Modeling

(L. Chen, et al. NeurIPS 2021)

Jihyeong Jeon
Seoul National University

Overview

- Problem Definition
- Proposed method
- Experiments

Outline

- **Problem Definition**
- Proposed Method
- Experiments

Problem Definition

- Given the historical interaction with the environment T, find the optimal parametrized policy π_θ, such that maximizes the return R
  - where, $\mathrm{T}_t^{T'} = \{S_t, A_t, r_t, S_{t+1}, A_{t+1}, r_{t+1}, \dots, S_{T'}, A_{T'}, r_{T'}\}$, parametrized policy $\pi_{\theta_t}(A_t|S_t) = P[A_t|S_t, \theta_t]$ which follows parametric distribution and return $R_t = \mathbb{E}_\theta[r_t|S = s_t]$

Outline

- Problem Definition
- **Proposed Method**
- Experiments

Proposed Method

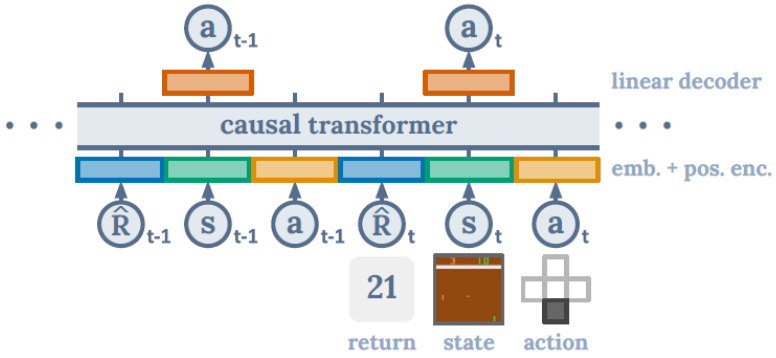
- Why Sequence Model over traditional RL algorithms?
  - It could ease the ‘Credit Assignment’ problem
    - No need of discount factor (hyperparameter) through self-attention
    - It also helps understanding long-term dependency
  - It has natural objective function which helps struggling in overestimation and error propagation

Proposed Method

- Trajectory Representation
  - Similar to the conventional RL setting, it consists of state, action, rewards
    - $\tau = \widehat{R_1}, s_1, a_1, \widehat{R_2}, s_2, a_2, \ldots, \widehat{R_T}, s_T, a_T$
      - where, $\widehat{R_t}$ is 'return-to-go' and s_t and a_t are state and action at time-step t respectively
    - The return-to-go (accumulated reward from time-step t to T),
      - $\widehat{R_t} = \sum_{t'=t}^{T} r_{t'}$

Proposed Method

- Autoregressive
  - K-time steps sampled from trajectory τ will be the input
  - Sampling an action from an inference
    - Which will be used as a next action
- Transformer
  - Each $\widehat{R_t}, s_t, a_t$ will be projected to embedding dimension
  - Positional embedding and layer normalization
  - Output vector of state embedding of decoder for next action



Outline

- Problem Definition
- Proposed Method
- **Experiments**

Experiments

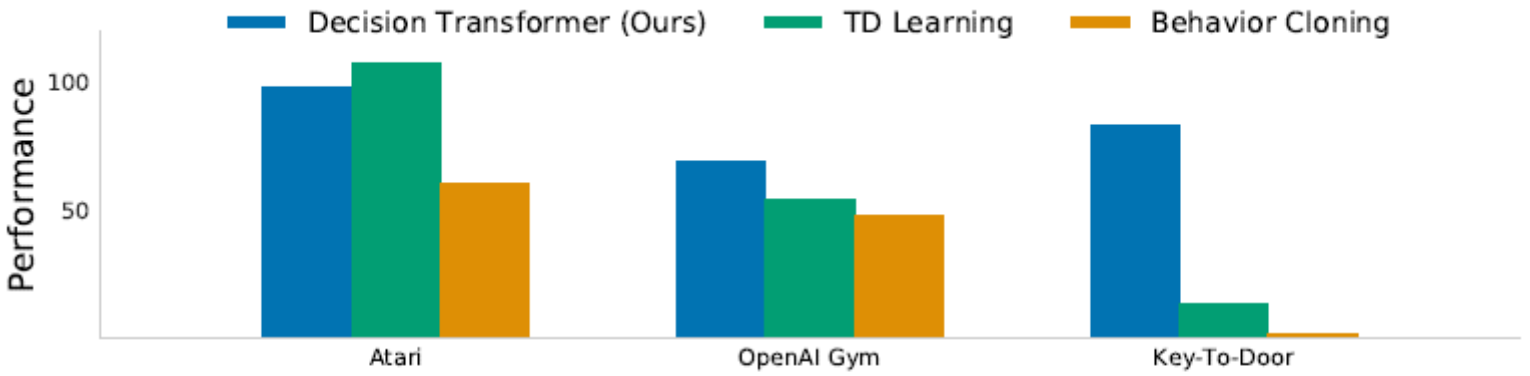


Figure 3: Results comparing Decision Transformer (ours) to TD learning (CQL) and behavior cloning across Atari, OpenAI Gym, and Minigrid. On a diverse set of tasks, Decision Transformer performs comparably or better than traditional approaches. Performance is measured by normalized episode return (see text for details).

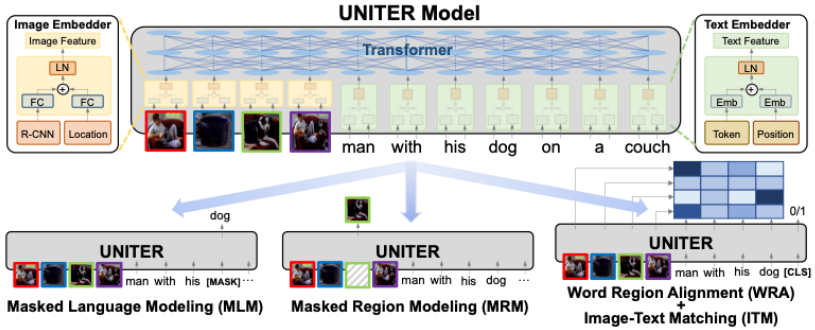
Thank you !

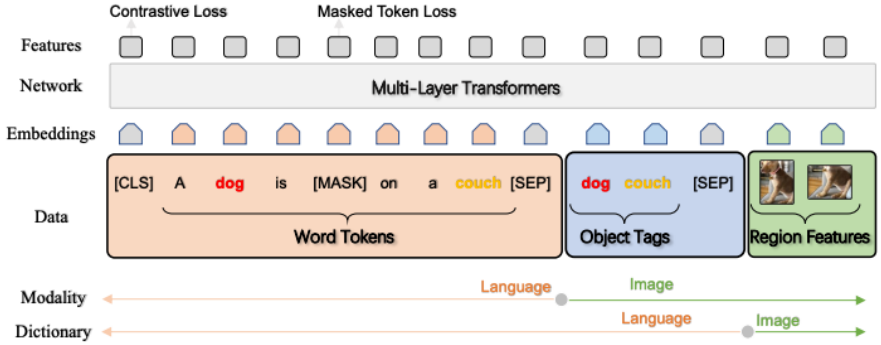
Align before Fuse: Vision and Language Representation Learning with Momentum Distillation

NeurIPS 2021

MinJoon Jung

Introduction





- Most existing methods employ a transformer-based multimodal encoder to jointly model visual tokens and word tokens. Because the visual tokens and word tokens are unaligned, it is challenging for the multimodal encoder to learn image-text interactions.

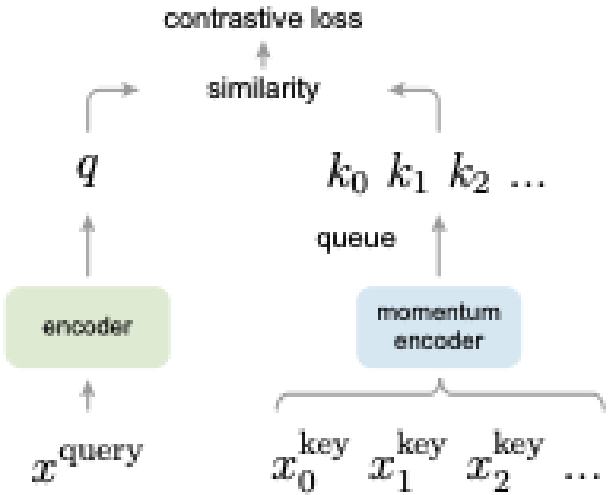
- Previous works [1], [2] rely on pre-trained object detectors to extract region-based image features, and employ a multimodal encoder to fuse the image features with word tokens

[1] UNITER: UNiversal Image-TExt Representation Learning
[2] Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks

Introduction

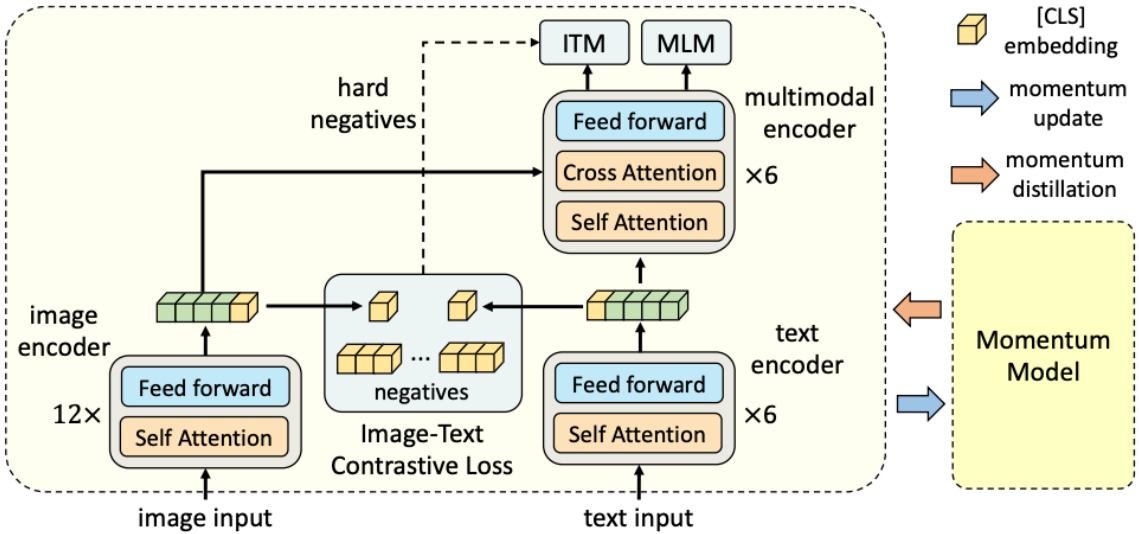
1. Vision-and Language Pre-training (VLP) suffers from several limitations
    a. Multimodal encoder is hard to learn their interactions (Vision - Language)
    b. Cost of object detector
    c. Image-text datasets are noisy, existing pre-training objectives (MLM) may overfit to the noisy text.
2. We propose ALign BEfore Fuse (ALBEF), a new VLP framework to address these limitations. We introduce an intermediate image-text contrastive (ITC) loss on representations from the unimodal encoders.
3. To improve learning under noisy supervision, we propose Momentum Distillation (MoD), a simple method which enables the model to leverage a larger uncurated web dataset.
4. We demonstrate the effectiveness of ALBEF on various downstream V+L tasks including image-text retrieval, visual question answering, visual reasoning, visual entailment, and weakly-supervised visual grounding.

Background



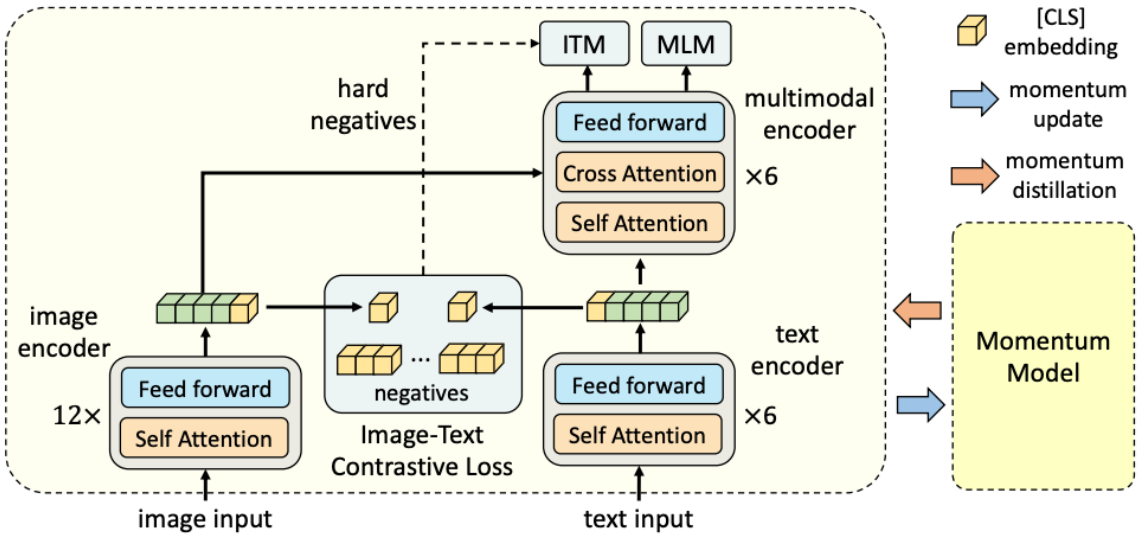
Momentum Contrast for Unsupervised Visual Representation Learning

ALBEF - model architecture



1. Image encoder : 12-layer visual transformer ViT-B/16
2. Text encoder : 6-layer BERT base model
3. Multimodal encoder : 6-layer BERT base model

ALBEF - Pre-training Objectives

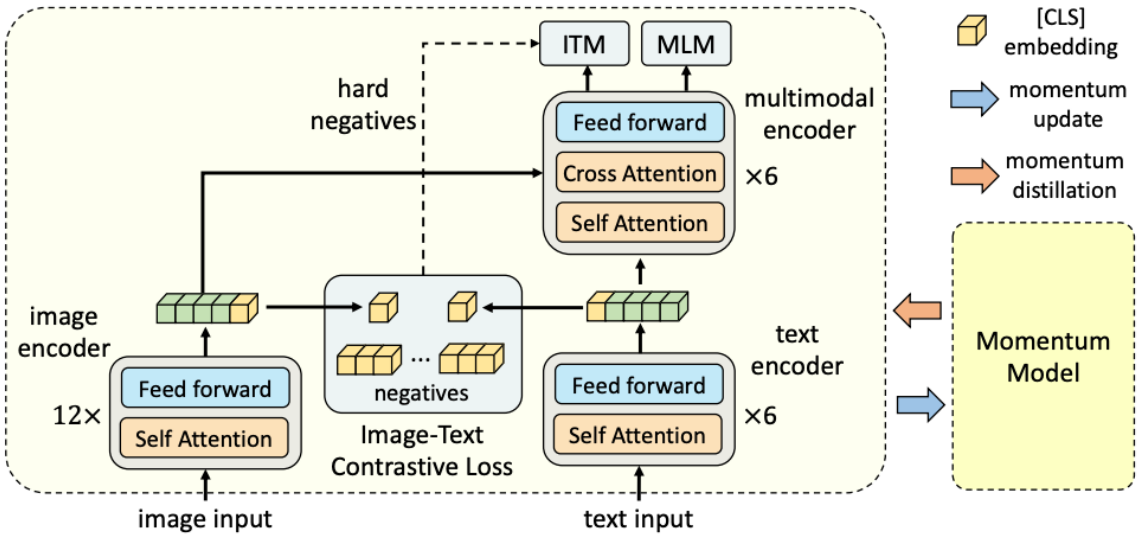


1. Image-text contrastive learning (ITC)
2. Masked language modeling (MLM)
3. Image-text matching (ITM) with hard negative mining

Momentum Distillation

- The image-text pairs used for pre-training are ***mostly collected from the web and they tend to be noisy***. Positive pairs are usually weakly-correlated. The text may contain words that are unrelated to the image, or the image may contain entities that are not described in the text.
  - For ITC learning, negative texts for an image may also match the image's content.
  - For MLM, there may exist other words different from the annotation that describes the image equally well (or better).
- However, the one-hot labels for **ITC and MLM penalize all negative predictions regardless of their correctness.**

ALBEF - Pre-training Objectives



1. Image-text contrastive learning (ITC)
2. Masked language modeling (MLM)
3. Image-text matching (ITM) with hard negative mining

"polar bear in the [MASK]"

GT: wild
Top-5 pseudo-targets:
1. zoo
2. pool
3. water
4. pond
5. wild

"a man [MASK] along a road in front of nature in summer"

GT: standing
Top-5 pseudo-targets:
1. walks
2. walking
3. runs
4. running
5. goes

"a [MASK] waterfall in the deep woods"

GT: remote
Top-5 pseudo-targets:
1. small
2. beautiful
3. little
4. secret
5. secluded

GT: breakdown of the car on the road
Top-5 pseudo-targets:
1. young woman get out of the car near the road
2. a woman inspects her damaged car under a tree
3. a woman looking into a car after locking her keys inside
4. young woman with a broken car calling for help
5. breakdown of the car on the road

GT: the harbor a small village
Top-5 pseudo-targets:
1. the harbour with boats and houses
2. replica of the sailing ship in the harbour
3. ships in the harbor of the town
4. the harbor a small village
5. boats lined up alongside the geographical feature category in the village

Figure 2: Examples of the pseudo-targets for MLM (1st row) and ITC (2nd row). The pseudo-targets can capture visual concepts that are not described by the ground-truth text (*e.g.* "beautiful waterfall", "young woman").

Downstream V+L Tasks

1. Image-Text Retrieval
2. Visual Entailment
3. Visual Question Answering
4. Natural Language for Visual Reasoning
5. Visual Grounding

Conclusion

- This paper proposes ALBEF, a new framework for vision-language representation learning.

- ALBEF first aligns the unimodal image representation and text representation before fusing them with a multimodal encoder

- Compared to existing methods, ALBEF offers better performance and faster inference speed on multiple downstream V+L tasks.

Swin Transformer : Hierarchical Vision Transformer using Shifted Windows

Yungi Jeong

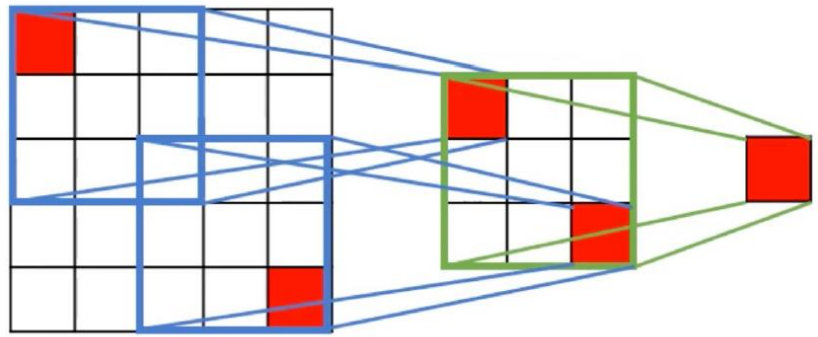
2021. 11. 19.

SEOUL NATIONAL UNIVERSITY
NUMERICAL COMPUTING & IMAGE ANALYSIS LAB

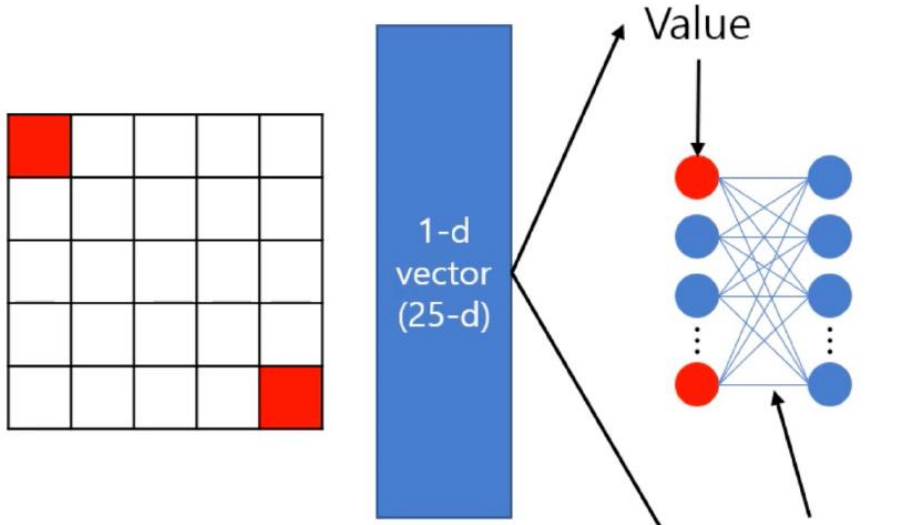
Introduction

Vision Transformer (ViT)

- **Concept**
  - Vision Transformer is a novel method in computer vision task **without CNN-like architectures**.
  - CNN extracts image features using a kernel consecutively, while ViT extracts them from a **self-attention** among all image patches.



CNN

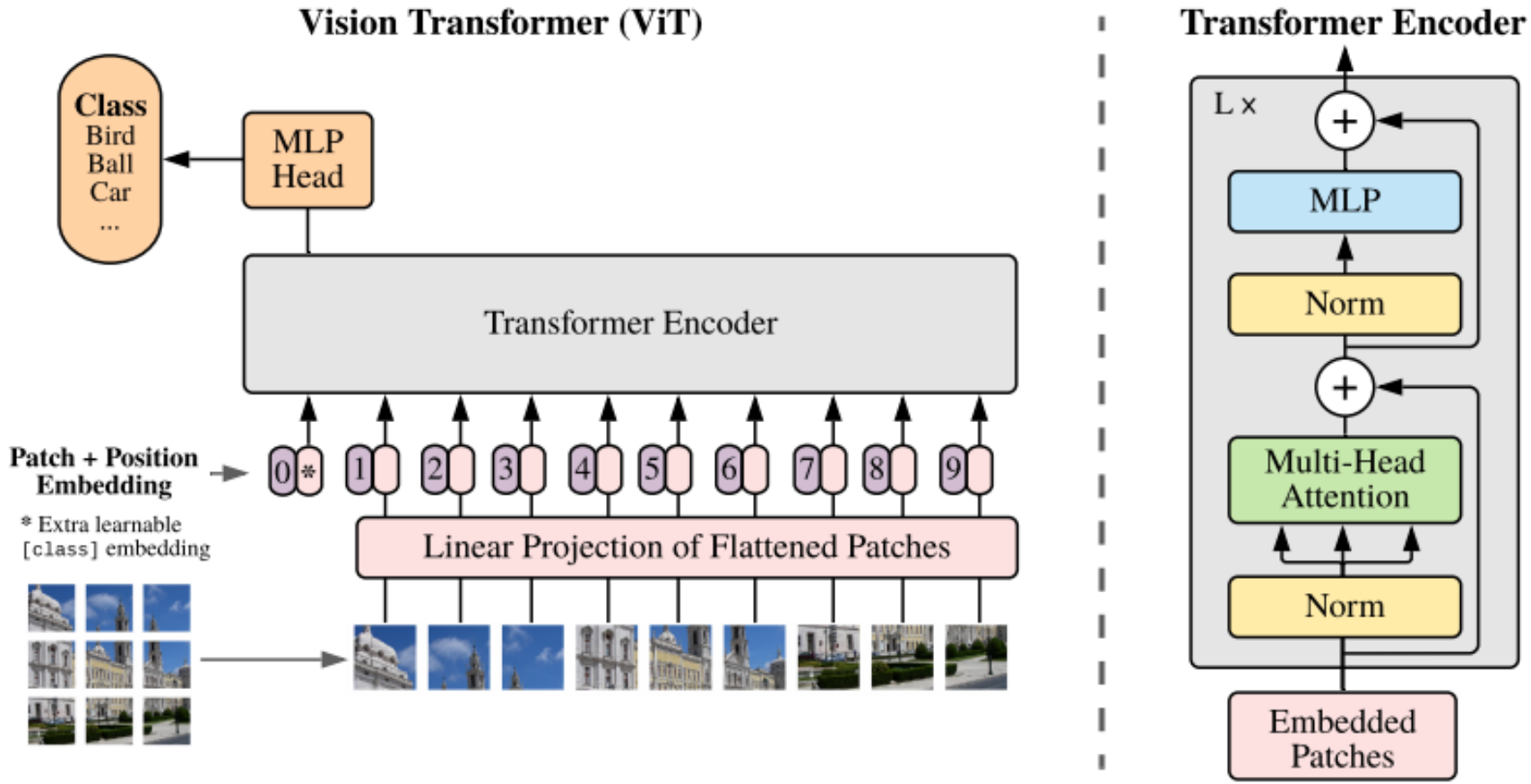


Transformer

Introduction

Vision Transformer (ViT)

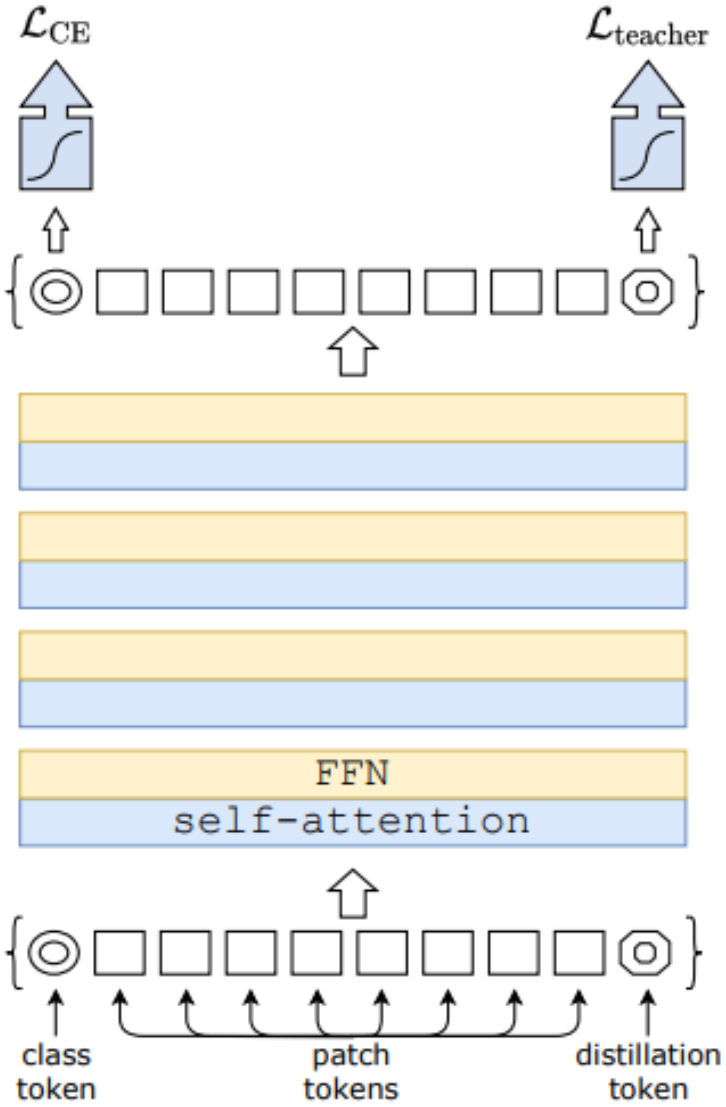
- Architecture



Introduction

Vision Transformer (ViT)

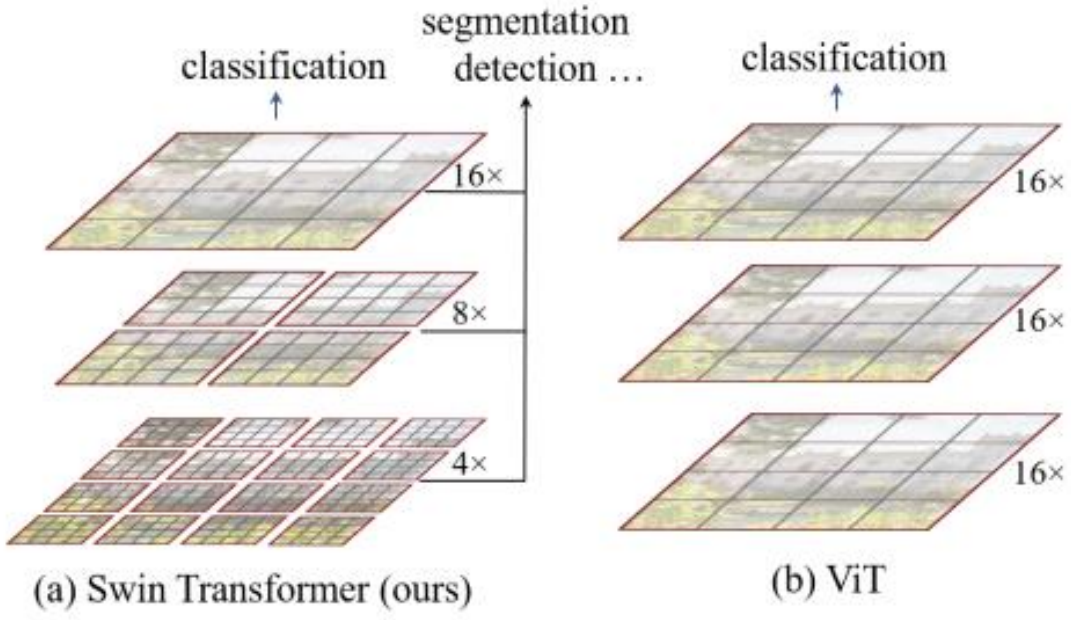
- **Limitation and Improvement**
  - ViT has many parameters compare to CNN architectures, which yields high computational cost.
  - ViT requires extremely large datasets and much longer training time to achieve better accuracy than CNN.
  - Data efficient image Transformer(**DeiT**) overcomes larger dataset and training time issues using **Knowledge Distillation**, adopting a CNN as the teacher model.



Swin Transformer

Introduction

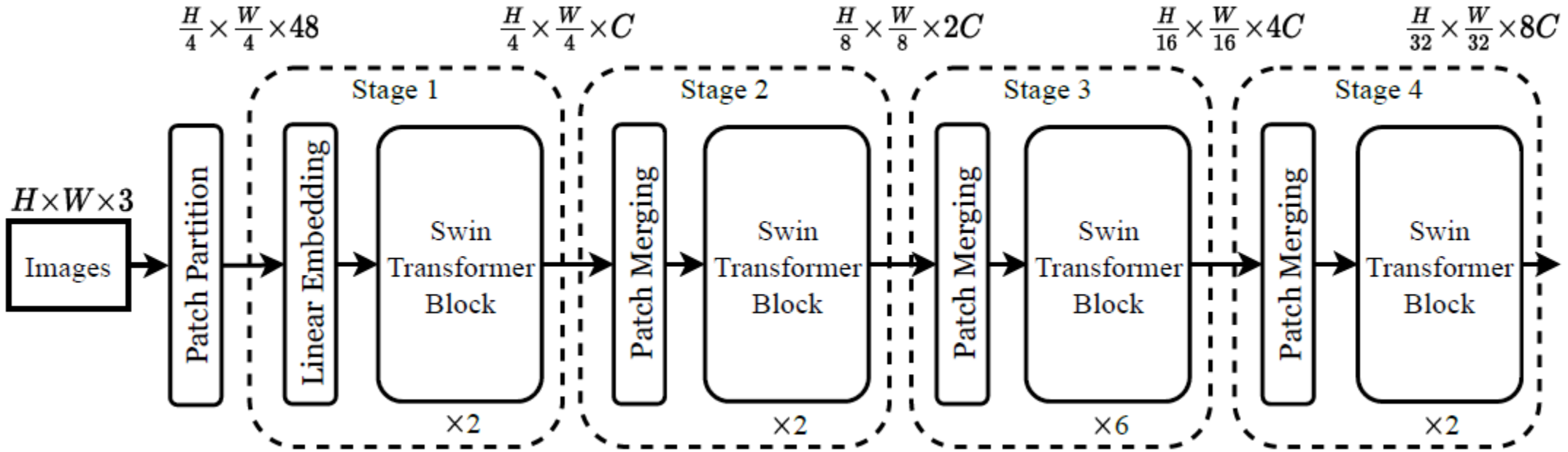
- **Hierarchical Vision Transformer using Shifted *Win*dows**
  - Hierarchical Transformer architecture whose representation is computed with **shifted windows**.
  - Swin Transformer reduces model parameters and computational complexity by applying **self-attention locally** within non-overlapping windows.
  - Unlike ViT, Swin Transformer can be exploited as a **backbone** for various vision tasks.



Swin Transformer

Model

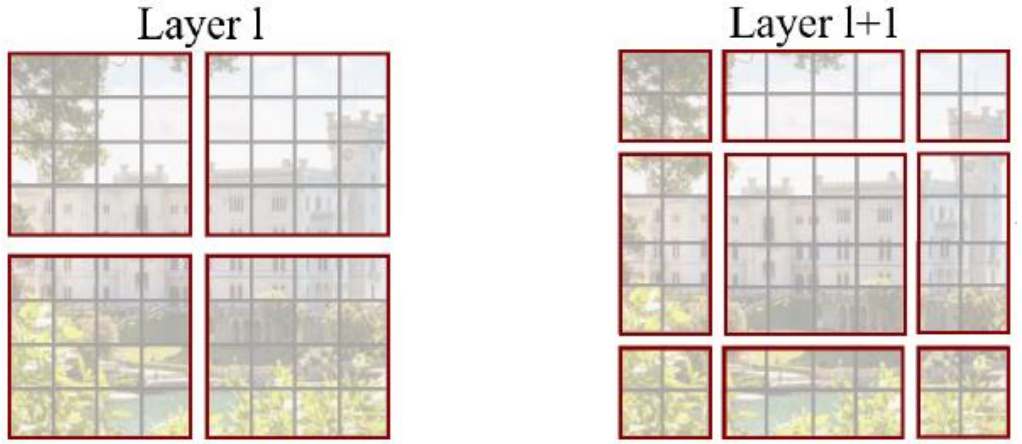
- **Architecture**

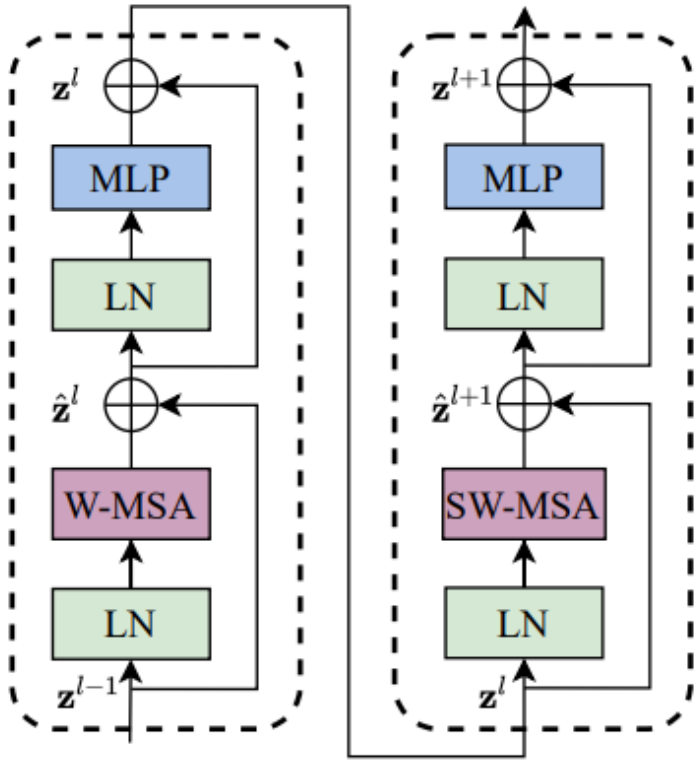


Swin Transformer

Model

- **Shifted Window based Self-Attention**
  - In each stage, a single or multiple pairs of transformer blocks are placed.
  - The first block locally computes self-attention among image patches in each window at a multi-head self-attention module.
  - The second block shifts window partitions and computes self -attention in shifted partitions, taking **cross-window connections**.





Swin Transformer

Experiments

(a) Regular ImageNet-1K trained models

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
ViT-B/16 [20]	384^2	86M	55.4G	85.9	77.9
ViT-L/16 [20]	384^2	307M	190.7G	27.3	76.5
DeiT-S [63]	224^2	22M	4.6G	940.4	79.8
DeiT-B [63]	224^2	86M	17.5G	292.3	81.8
DeiT-B [63]	384^2	86M	55.4G	85.9	83.1
Swin-T	224^2	29M	4.5G	755.2	81.3
Swin-S	224^2	50M	8.7G	436.9	83.0
Swin-B	224^2	88M	15.4G	278.1	83.5
Swin-B	384^2	88M	47.0G	84.7	84.5

(b) ImageNet-22K pre-trained models

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
R-101x3 [38]	384^2	388M	204.6G	-	84.4
R-152x4 [38]	480^2	937M	840.5G	-	85.4
ViT-B/16 [20]	384^2	86M	55.4G	85.9	84.0
ViT-L/16 [20]	384^2	307M	190.7G	27.3	85.2
Swin-B	224^2	88M	15.4G	278.1	85.2
Swin-B	384^2	88M	47.0G	84.7	86.4
Swin-L	384^2	197M	103.9G	42.1	87.3

(a) Various frameworks

Method	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	#param.	FLOPs	FPS
Cascade Mask R-CNN	R-50	46.3	64.3	50.5	82M	739G	18.0
	Swin-T	**50.5**	**69.3**	**54.9**	86M	745G	15.3
ATSS	R-50	43.5	61.9	47.0	32M	205G	28.3
	Swin-T	**47.2**	**66.5**	**51.3**	36M	215G	22.3
RepPointsV2	R-50	46.5	64.6	50.3	42M	274G	13.6
	Swin-T	**50.0**	**68.5**	**54.2**	45M	283G	12.0
Sparse R-CNN	R-50	44.5	63.4	48.2	106M	166G	21.0
	Swin-T	**47.9**	**67.3**	**52.3**	110M	172G	18.4

(b) Various backbones w. Cascade Mask R-CNN

	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	param	FLOPs	FPS
DeiT-S†	48.0	67.2	51.7	41.4	64.2	44.3	80M	889G	10.4
R50	46.3	64.3	50.5	40.1	61.7	43.4	82M	739G	18.0
Swin-T	**50.5**	**69.3**	**54.9**	**43.7**	**66.6**	**47.1**	86M	745G	15.3
X101-32	48.1	66.5	52.4	41.6	63.9	45.2	101M	819G	12.8
Swin-S	**51.8**	**70.4**	**56.3**	**44.7**	**67.9**	**48.5**	107M	838G	12.0
X101-64	48.3	66.4	52.3	41.7	64.0	45.1	140M	972G	10.4
Swin-B	**51.9**	**70.9**	**56.5**	**45.0**	**68.4**	**48.7**	145M	982G	11.6

Reference

[1] Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." *arXiv preprint arXiv:2010.11929* (2020).

[2] Touvron, Hugo, et al. "Training data-efficient image transformers & distillation through attention." *International Conference on Machine Learning*. PMLR, 2021.

[3] Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. "Distilling the knowledge in a neural network." *arXiv preprint arXiv:1503.02531* (2015).

[4] Liu, Ze, et al. "Swin transformer: Hierarchical vision transformer using shifted windows." *arXiv preprint arXiv:2103.14030* (2021).

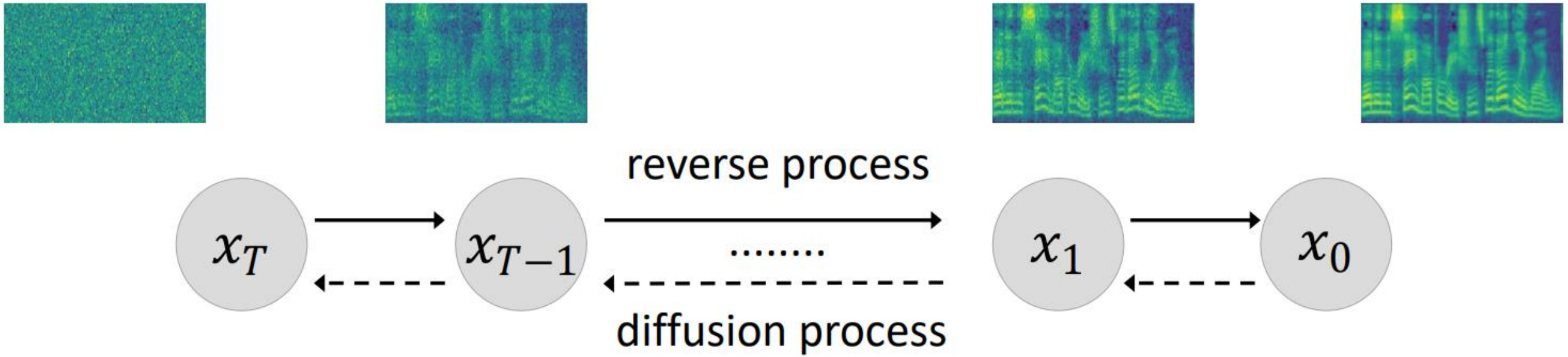
Diff-TTS: A Denoising Diffusion Model for Text-to-Speech (Interspeech 2021)

인공지능 협동과정
채윤기

Contribution

- Non-autoregressive TTS에 denoising diffusion probabilistic model(DDPM)을 적용한 첫 논문
- Parameter의 수가 Tacotron2나 Glow-TTS의 반 밖에 되지 않음
- Accelerated sampling을 적용하여 inference speed를 높였음
- “Temperature term”을 이용하여 pitch variability를 조정할 수 있음

Diff-TTS

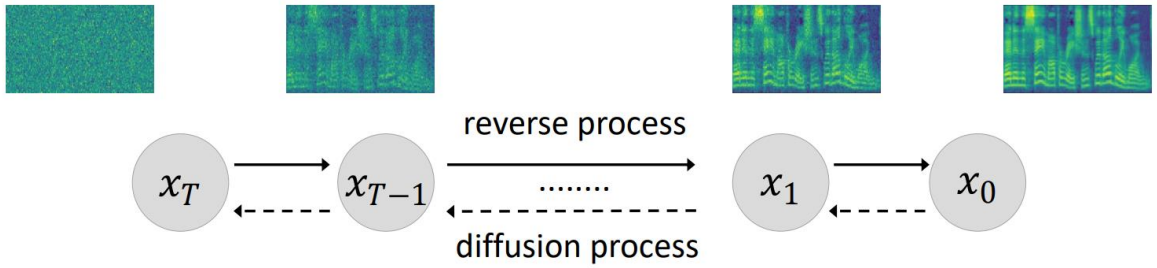


Diff-TTS

- $\alpha, \beta, \eta, \ldots$: hyperparameters.
- c : Text

- Diffusion process : $q(x_{1:T}|x_0, c)$ ← hyperparameter로 주어진대로 Markov chain.

- Reverse process: $p_\theta(x_{0:T-1}|x_T, c)$ ← 학습할 모델



$$q(x_t|x_{t-1}, c) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

$$q(x_1 \ldots, x_T|x_0, c) = \prod_{t=1}^{T} q(x_t|x_{t-1})$$

$$p_\theta(x_0 \ldots, x_{T-1}|x_T, c) = \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t, c)$$

Objective

- 최종 목적: Model $p_\theta(x_0|c)$를 $q(x_0|c)$(mel-spectrogram)에 근사
- $maximize_\theta\ \mathbb{E}_{logq(x_0|c)}[logp_\theta(x_0|c)]$
- 단, $p_\theta(x_0|c)$는 intractable. 따라서, variational lower bound를 계산함. (VAE의 ELBO식 이용하여 유도가능)
- Lower bound는 마이너스를 붙인 식의 upper bound가 되어 이를 minimize하게 됨.
- 최종적인 objective function은 다음과 같이 유도됨.

$$\min_\theta L(\theta) = \mathbb{E}_{x_0,\epsilon,t}\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t, c)\|_1$$

- $\alpha_t = 1 - \beta_t, \quad \bar{\alpha}_t = \prod_{t'=1}^{t} \alpha_{t'}$
- $\epsilon \sim \mathcal{N}(0, I)$
- $t = 1, \ldots, T$

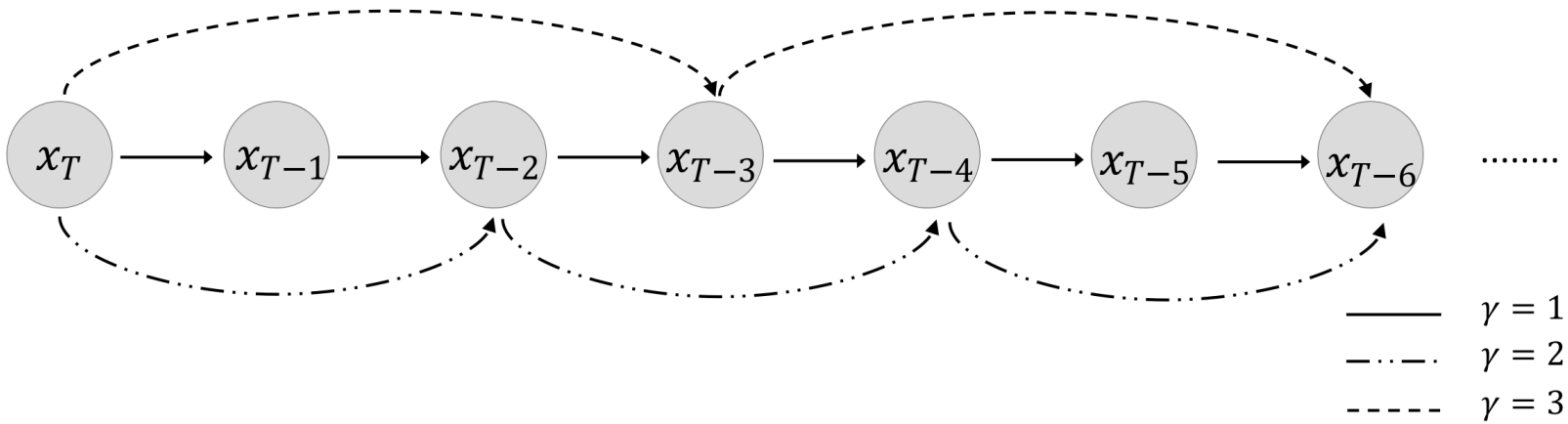
Inference

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t, c)) + \sigma_t z_t$$

- Objective에서 추정했던 노이즈 ϵ_θ를 iterative하게 time-step마다 제거하는 $z_t \sim \mathcal{N}(0, I)$
- $\sigma_t = \eta\sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t}$
-

- η: Temperature term. Variance의 scaling factor.

Accelerated Sampling



- Decimation factor γ만큼 skip하며 inference 하도록 함.
- Skip하더라도 sample quality가 저하되지 않도록 할 수 있음.
- 새로운 path를 $\tau = [\tau_1, \tau_2, \ldots, \tau_M](M < T)$로 놓으면, $i > 1$에 대해 sampling을 다음과 같이 진행함.

$$x_{\tau_{i-1}} = \sqrt{\bar{\alpha}_{\tau_{i-1}}}\left(\frac{x_{\tau_i} - \sqrt{1-\bar{\alpha}_{\tau_i}}\epsilon_\theta(x_{\tau_i}, \tau_i, c)}{\sqrt{\bar{\alpha}_{\tau_i}}}\right)$$

$$+ \sqrt{1 - \bar{\alpha}_{\tau_{i-1}} - \sigma_{\tau_i}^2}\,\epsilon_\theta(x_{\tau_i}, \tau_i, c) + \sigma_{\tau_i} z_{\tau_i}$$

$$\text{where } \sigma_{\tau_i} = \eta\sqrt{\frac{1-\bar{\alpha}_{\tau_{i-1}}}{1-\bar{\alpha}_{\tau_i}}\beta_{\tau_i}}$$

Accelerated Sampling

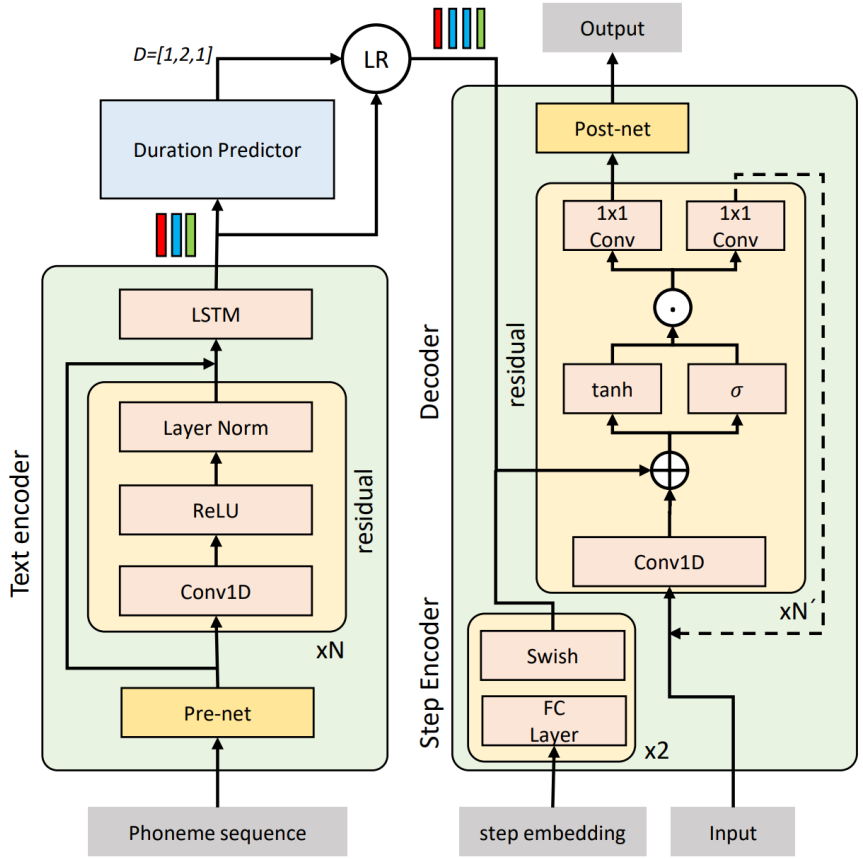
$$x_{\tau_{i-1}} = \sqrt{\bar{\alpha}_{\tau_{i-1}}}\left(\frac{x_{\tau_i} - \sqrt{1-\bar{\alpha}_{\tau_i}}\epsilon_\theta(x_{\tau_i}, \tau_i, c)}{\sqrt{\bar{\alpha}_{\tau_i}}}\right) \qquad \text{where } \sigma_{\tau_i} = \eta\sqrt{\frac{1-\bar{\alpha}_{\tau_{i-1}}}{1-\bar{\alpha}_{\tau_i}}\beta_{\tau_i}}$$

$$+ \sqrt{1-\bar{\alpha}_{\tau_{i-1}} - \sigma_{\tau_i}^2}\,\epsilon_\theta(x_{\tau_i}, \tau_i, c) + \sigma_{\tau_i} z_{\tau_i}$$

- 마지막 x_0의 경우, x_{τ_1}로부터 다음과 같이 구할 수 있음

$$x_0 = \frac{x_{\tau_1} - \sqrt{1-\bar{\alpha}_{\tau_1}}\epsilon_\theta(x_{\tau_1}, \tau_1, c)}{\sqrt{\bar{\alpha}_{\tau_1}}}$$

Model Architecture



- Text Encoder
  Phoneme sequence로부터 contextual information추출하여 duration prediction으로 전달.

- Duration Predictor / Length Regulator
  Phoneme과 mel-spectrogram의 길이를 맞춤

- Step Encoder
  Step t 를 embedding 하여 Decoder에 condition으로서 전달

- Decoder
  t-번째 step의 latent variable로부터 Gaussian noise 추정

Experiments

Table 1: *The Mean Opinion Score (MOS) of single speaker TTS models with 95% confidence intervals.*

Method	**5-scale MOS**
GT	4.541 ± 0.057
GT(Mel + HiFiGAN)	4.323 ± 0.060
Tacotron2	4.006 ± 0.072
Glow-TTS	4.160 ± 0.070
Diff-TTS(T=400, $\gamma = 1$)	$\mathbf{4.337 \pm 0.064}$
Diff-TTS(T=400, $\gamma = 7$)	4.223 ± 0.066
Diff-TTS(T=400, $\gamma = 21$)	4.135 ± 0.070
Diff-TTS(T=400, $\gamma = 57$)	4.091 ± 0.067

Table 3: *The comparison of RTF in mel spectrogram synthesis. RTF denotes the real-time factor, that is the time (in seconds) required for the system to synthesize one second waveform.*

Method	**RTF**
Tacotron2	0.117
Glow-TTS	**0.008**
Diff-TTS(T=400)	
Normal diffusion sampling	1.744
Accelerated sampling($\gamma = 7$)	0.258
Accelerated sampling($\gamma = 21$)	0.090
Accelerated sampling($\gamma = 57$)	**0.035**

Diff-TTS (jmhxxi.github.io)

Reference

- (Figures are from)M.Jeong, H.Kim, J.Sung, B.Choi, and N.Kim, “Diff-TTS: A Denoising Diffusion Model for Text-to-Speech,” Interspeech 2021.
- P.Dhariwal, A.Nichol, “Diffusion Models Beat GANS on Image Synthesis,” arXiv preprint arXiv:2105.05233 2021.
- J.Ho, A.Jain, and P.Abbeel, “Denoising Diffusion Probabilistic Models,” arXiv preprint arXiv:2006.11239, 2020.
- J.Song, C.Meng, and S.Ermon, “Denoising diffusion implicit models,” arXiv preprint arXiv:2010.02502, 2020.

Learning Transferable Visual Models From Natural Language Supervision :CLIP (ICML 2021)

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh , Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pa mela Mishkin, Jack Clark, Gretchen Krueger Ilya Sutskever (OpenAI)

Presenters : 한동훈

Method

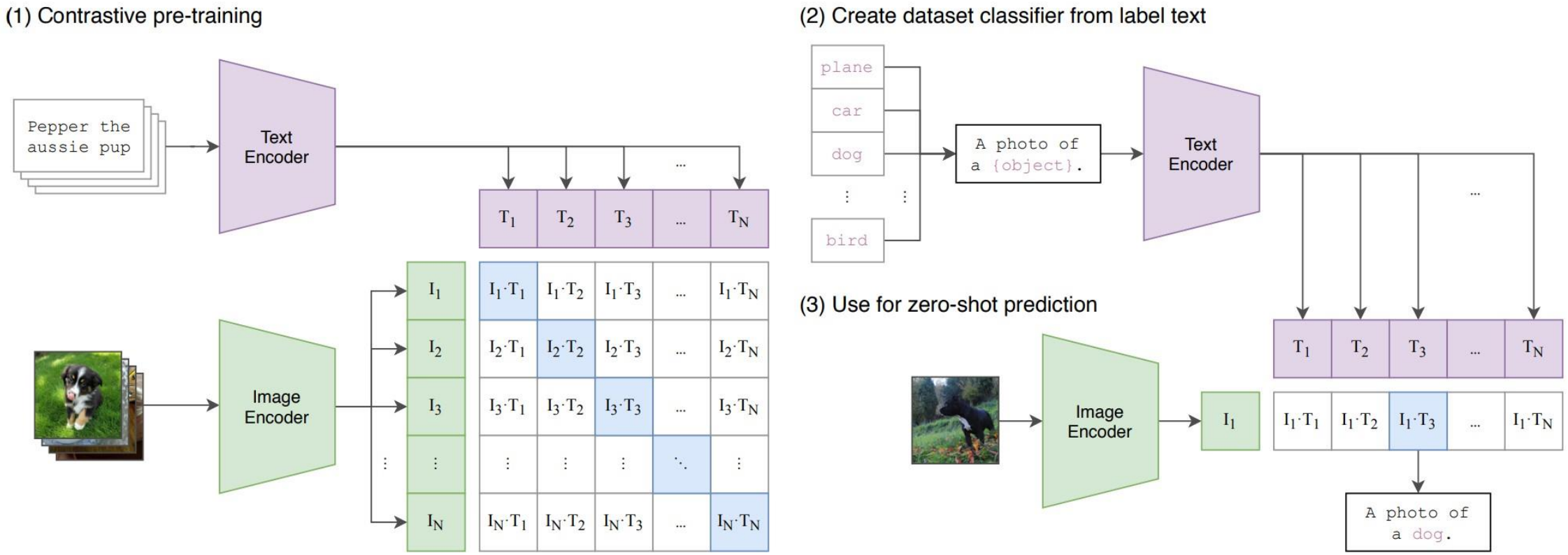
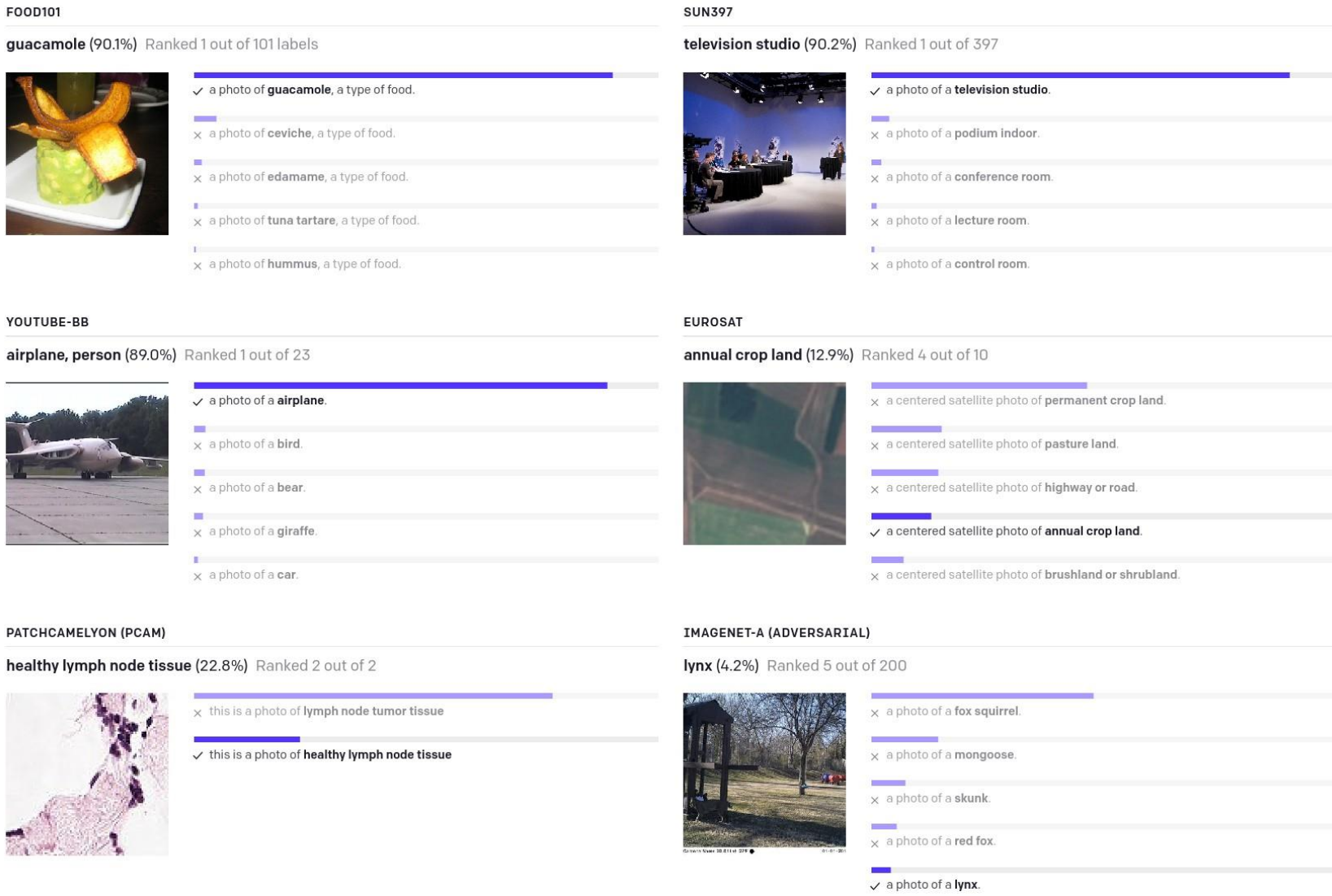


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

Examples (supplementary material)



Approach (supplementary material)

Selecting an Efficient Pre-Training Method

- Given batch of N pairs: NxN possible pairs
- CLIP learns multi-modal embedding space by jointly trai ning image encoder and text to maximize the cosine simil arity
- Train CLIP from scratch
- Linearly project the extracted feature to embedding space

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

A Practical Bayesian Framework for Structural Model Updating and Prediction_review

건설환경공학부
2020-25797
현수민

Introduction

- Vibration-based structural health monitoring of target structure generally relies on a reasonable and accurate finite-element model for damage parameter identification

- For the FE model-based SHM procedure, the accuracy of the FE model is essential for its successful implementation

- To ensure the accuracy of the established FE model of the target structure, the initial model should be calibrated or updated through adjusting appropriate model parameters with the observed data

- Most of the previous research works are based on deterministic methods

- Deterministic FE model updating aims to find the optimal parameters of the numerical model to obtain the best fit between the model output and the measured data

- This is usually achieved by describing the problem as a constrained optimization problem, in which the goal is to minimize the difference between the calculated data and the observed one

- However, model updating is essentially an inverse problem, and the obtained results are greatly affected by the incomplete measurement, measurement noise, modeling errors, and so on

Proposed Methodology-1

=> This paper proposes a practical framework for structural model updating and prediction by utilizing the modal parameters based on the Bayesian regularization

- Structural Model Updating Based on the Bayesian Regularization

- Simultaneously updating the model parameters together with the three hyperparameters
- Predicting a natural frequency vector

$$p(\boldsymbol{\theta}|\mathbb{D}_N, \alpha, \beta, \gamma, \mathbb{M}) = \frac{p(\mathbb{D}_N|\boldsymbol{\theta}, \beta, \gamma, \mathbb{M})\, p(\boldsymbol{\theta}|\alpha, \mathbb{M})}{p(\mathbb{D}_N|\alpha, \beta, \gamma, \mathbb{M})} \qquad (10)$$

Posterior distribution of uncertain model parameters conditional on the measurement data

$$\begin{aligned}\ln p(\mathbb{D}_N|\alpha, \beta, \gamma, \mathbb{M}) \simeq\ & \frac{N_\theta}{2}\ln\alpha - \frac{\alpha}{2}\|\boldsymbol{\theta}_{\mathrm{MAP}} - \boldsymbol{\theta}_0\|^2 + \frac{NN_t}{2}\ln\beta \\ & - \frac{\beta}{2}\sum_{n=1}^{N}\|\mathbf{f}(\boldsymbol{\theta};\mathbb{M}) - \mathbf{f}^{(n)}\|^2 + \frac{NN_{mt}}{2}\ln\gamma \\ & - \frac{\gamma}{2}\sum_{n=1}^{N}\sum_{j=1}^{N_t}\left(1 - \kappa_j\left(\boldsymbol{\theta};\boldsymbol{\psi}_j^{(n)},\mathbb{M}\right)\right) \\ & + \frac{N_\theta}{2}\ln 2\pi - \frac{1}{2}\ln\ \det\mathbf{A}(\boldsymbol{\theta}_{\mathrm{MAP}};\alpha,\beta,\gamma,\mathbb{M}) \qquad (18)\end{aligned}$$

Taking the partial derivative of the log-evidence with respect to the three hyperparameters and setting them equal to zero

Proposed Methodology-summary

1. Initialize the model parameters $\boldsymbol{\theta} = \boldsymbol{\theta}_0$ and hyperparameters $\alpha = 1{,}000$, $\beta = 1$, and $\gamma = 1$. Set the lower and upper bounds of $\boldsymbol{\theta}$ as $\mathbf{l}_\theta$ and $\mathbf{u}_\theta$, respectively.
2. At the kth iteration step, extract the optimal values of hyperparameters $\alpha_{\text{MAP}}^{(k-1)} = \alpha(\boldsymbol{\theta}_{\text{MAP}}^{(k-1)})$, $\beta_{\text{MAP}}^{(k-1)} = \beta(\boldsymbol{\theta}_{\text{MAP}}^{(k-1)})$, and $\gamma_{\text{MAP}}^{(k-1)} = \gamma(\boldsymbol{\theta}_{\text{MAP}}^{(k-1)})$ obtained from the $(k-1)$th iteration step.
3. Solve the problem of nonlinear bound-constrained least-squares minimization in Eq. (23) with the trust-region-reflective algorithm to obtain the current $\boldsymbol{\theta}_{\text{MAP}}^{(k)}$, the full-length residual vector $\mathbf{r}(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \alpha_{\text{MAP}}^{(k-1)}, \beta_{\text{MAP}}^{(k-1)}, \gamma_{\text{MAP}}^{(k-1)}, \mathbb{M})$, and the corresponding Jacobian matrix $\mathbf{J}(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \alpha_{\text{MAP}}^{(k-1)}, \beta_{\text{MAP}}^{(k-1)}, \gamma_{\text{MAP}}^{(k-1)}, \mathbb{M})$.
4. Extract the three residuals $\mathbf{r}_1(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \mathbb{M})$, $\mathbf{r}_2(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \mathbb{M})$, and $\mathbf{r}_3(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \mathbb{M})$ at the current $\boldsymbol{\theta}_{\text{MAP}}^{(k)}$ through Eqs. (36)–(38), respectively.
5. Extract the two Jacobian matrices $\mathbf{J}_1(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \mathbb{M})$ and $\mathbf{J}_2(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \mathbb{M})$ from Eqs. (33) and (34), respectively.
6. Calculate the approximated Hessian matrices $\mathbf{H}_1(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \mathbb{M})$ and $\mathbf{H}_2(\boldsymbol{\theta}_{\text{MAP}}^{(k)}; \mathbb{M})$ at $\boldsymbol{\theta}_{\text{MAP}}^{(k)}$ by using Eqs. (39) and (40).
7. Calculate the approximated inverse Hessian of the negative logarithm of the posterior distribution $\boldsymbol{\Sigma}\left(\boldsymbol{\theta}_{\text{MAP}}; \alpha_{\text{MAP}}^{(k-1)}, \beta_{\text{MAP}}^{(k-1)}, \gamma_{\text{MAP}}^{(k-1)}, \mathbb{M}\right)$ evaluated at $\boldsymbol{\theta}_{\text{MAP}}^{(k)}$ through Eq. (42).
8. Update the hyperparameters $\beta_{\text{MAP}}^{(k-1)}$, $\gamma_{\text{MAP}}^{(k-1)}$, and $\alpha_{\text{MAP}}^{(k-1)}$ sequentially through Eqs. (41), (43), and (44) to obtain $\beta_{\text{MAP}}^{(k)}$, $\gamma_{\text{MAP}}^{(k)}$, and $\alpha_{\text{MAP}}^{(k)}$ at current $\boldsymbol{\theta}_{\text{MAP}}^{(k)}$, respectively.
9. Repeat Steps 2–8 until the convergence criterion is met; i.e., the norm error of optimal model parameters between the current and previous iteration steps $\boldsymbol{\theta}_{\text{MAP}}^{(k)} - \boldsymbol{\theta}_{\text{MAP}}^{(k-1)}$ is less than a prescribed small ε.
10. Output the optimal values of model parameters $\boldsymbol{\theta}_{\text{MAP}}$, together with the three hyperparameters α_{MAP}, β_{MAP}, and γ_{MAP}.

Proposed Methodology-2

- Prediction of Structural Modal Parameters Based on the Updated Model

- Instead of utilizing the usual Bayesian treatment for hyperparameters, which involves marginalization of the posterior distribution over all possible values of the three hyperparameters

$$p(\boldsymbol{\theta}|\mathbb{D}_N, \mathbb{M}) \simeq \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\theta}_{\text{MAP}}, \mathbf{A}^{-1}(\boldsymbol{\theta}_{\text{MAP}}; \alpha_{\text{MAP}}, \beta_{\text{MAP}}, \gamma_{\text{MAP}}, \mathbb{M})) \tag{45}$$

$$p(\mathbf{f}|\mathbb{D}_N, \mathbb{M}) = \int p(\mathbf{f}|\boldsymbol{\theta}, \beta, \mathbb{M}) p(\boldsymbol{\theta}|\mathbb{D}_N, \mathbb{M}) \mathrm{d}\boldsymbol{\theta} \tag{46}$$

Natural frequencies

$$p(\boldsymbol{\psi}_j|\mathbb{D}_N, \mathbb{M}) = \int p(\boldsymbol{\psi}_j|\boldsymbol{\theta}, \gamma, \mathbb{M}) p(\boldsymbol{\theta}|\mathbb{D}_N, \mathbb{M}) \mathrm{d}\boldsymbol{\theta} \tag{47}$$

Mode shapes

- Model predicted natural frequency and model shape vector on the uncertain model parameters are still intractable due to the nonlinearity of FE model functions

- Assumption : covariance of posterior distribution of model parameters is small (linear approximation)

- Using a Taylor series expansion of the FE model functions

$$p(\mathbf{f}|\boldsymbol{\theta}, \beta, \mathbb{M}) \simeq \mathcal{N}\Big(\mathbf{f}|\mathbf{y}_1(\boldsymbol{\theta}_{\text{MAP}}; \mathbb{M}) + \mathbf{J}_1(\boldsymbol{\theta}_{\text{MAP}}; \mathbb{M})(\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}}), \beta_{\text{MAP}}^{-1} \mathbf{I}_{N_r}\Big) \tag{48}$$

Case study

- Real life pedestrian steel bridge located in Xima Road, Wuhan, China

Fig. 1. Xima Road Pedestrian Bridge in Wuhan, PR China. (©2020 Baidu.)

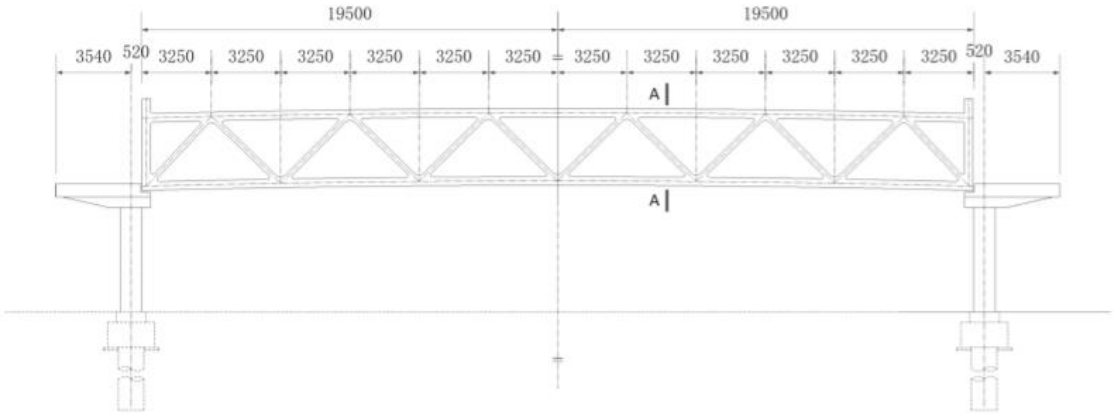


Fig. 2. Elevation view of the pedestrian bridge (unit:mm).

Table 1. Geometrical and material properties used in the FE model of the pedestrian steel bridge

Parameter descriptions	Initial values
Bridge span	39 m
Bridge width	4.3 m
Bridge height	3.4 m
Young's modulus of steel material	2.06×10^{11} N/m^2
Mass density of steel material	7.85×10^3 kg/m^3
Poisson's ratio of steel material	0.3
Outer dimension of top chord section (rectangular hollow)	0.4 × 0.3 m
Thickness of top chord section	0.012 m
Outer dimension of bottom chord section (rectangular hollow)	0.4 × 0.3 m
Thickness of bottom chord section	0.012 m
Outer dimension of diagonal chord (rectangular hollow)	0.25 × 0.3 m
Thickness of diagonal chord	0.012 m
Outer radius of strut section (circular hollow)	0.09 m
Thickness of strut section	0.012 m
Thickness of top panel of bridge deck	0.012 m
Thickness of bottom panel of bridge deck	0.012 m
Thickness of rubber pavement	0.02 m
Mass density of rubber pavement	1.5×10^3 kg/m^3

Case study

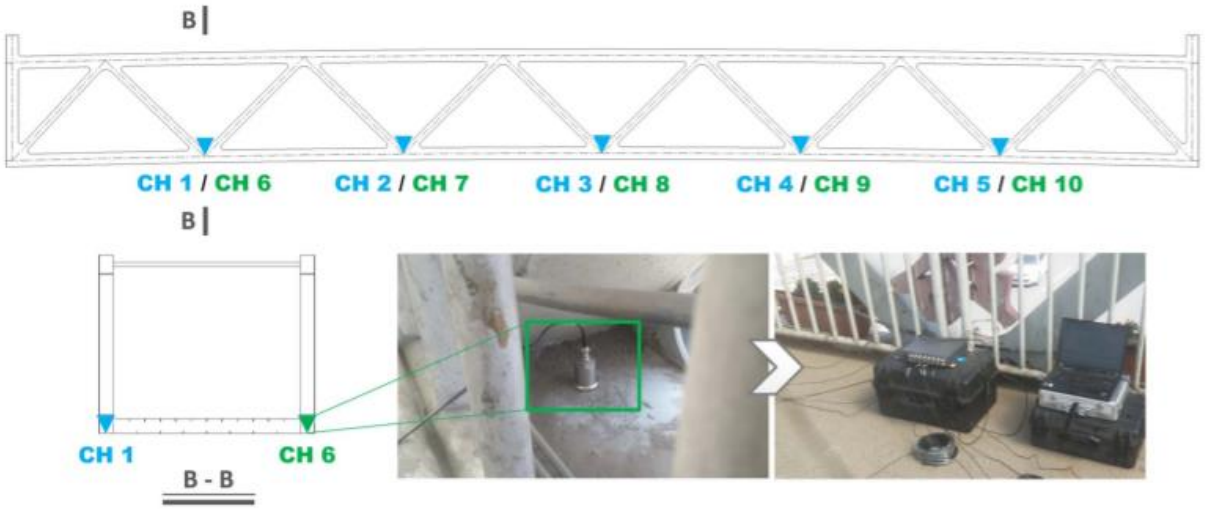


Fig. 4. Configuration of 10 measuring points in the field testing for the pedestrian bridge.

Table 2. Experimental natural frequencies and mode shapes of three data sets used in FE model updating

Data set number	Items	Mode 1	Mode 2	Mode 3	Mode 4
1	Natural frequencies (Hz)	4.8520	9.7053	13.4765	15.4848
	Mode shape components	0.1922	−0.1986	−0.3634	−0.5888
		0.3256	−0.3163	−0.3675	−0.1165
		0.4201	−0.4127	−0.0463	0.0649
		0.3546	−0.3452	0.3384	0.1306
		0.2460	−0.2528	0.4972	0.3381
		0.1997	0.2074	−0.3094	0.5382
		0.3274	0.3155	−0.3187	0.2446
		0.4139	0.4103	−0.0702	0.0530
		0.3555	0.3686	0.2386	−0.2173
		0.2226	0.2457	0.3315	−0.3235
2	Natural frequencies (Hz)	4.8512	9.7005	13.4635	15.4901
	Mode shape components	0.1919	−0.2024	−0.3666	−0.5395
		0.3258	−0.3199	−0.3689	−0.1156
		0.4203	−0.4074	−0.0551	−0.0116
		0.3552	−0.3416	0.3426	0.1620
		0.2426	−0.2626	0.5044	0.4862
		0.1993	0.2009	−0.3046	0.4461
		0.3284	0.3204	−0.3256	0.2374
		0.4150	0.4077	−0.0756	0.0398
		0.3553	0.3636	0.2316	−0.1824
		0.2221	0.2525	0.3108	−0.3777
3	Natural frequencies (Hz)	4.8421	9.7125	13.4595	15.5315
	Mode shape components	0.1926	−0.2031	−0.3763	−0.5257
		0.3254	−0.3193	−0.3695	−0.1742
		0.4194	−0.4066	−0.0521	−0.0160
		0.3547	−0.3412	0.3446	0.2016
		0.2440	−0.2571	0.5039	0.4185
		0.1981	0.2030	−0.3024	0.4776
		0.3288	0.3241	−0.3266	0.2450
		0.4153	0.4080	−0.0758	0.0654
		0.3559	0.3655	0.2253	−0.2214
		0.2218	0.2503	0.3033	−0.3686

3 Data sets

Case study

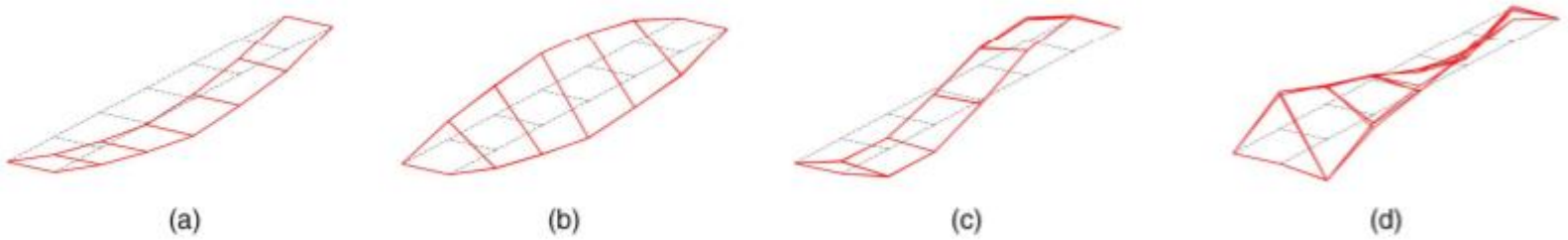


Fig. 5. Overlapped display of three sets of experimental mode shapes identified from field testing: (a) Mode 1; (b) Mode 2; (c) Mode 3; and (d) Mode 4.

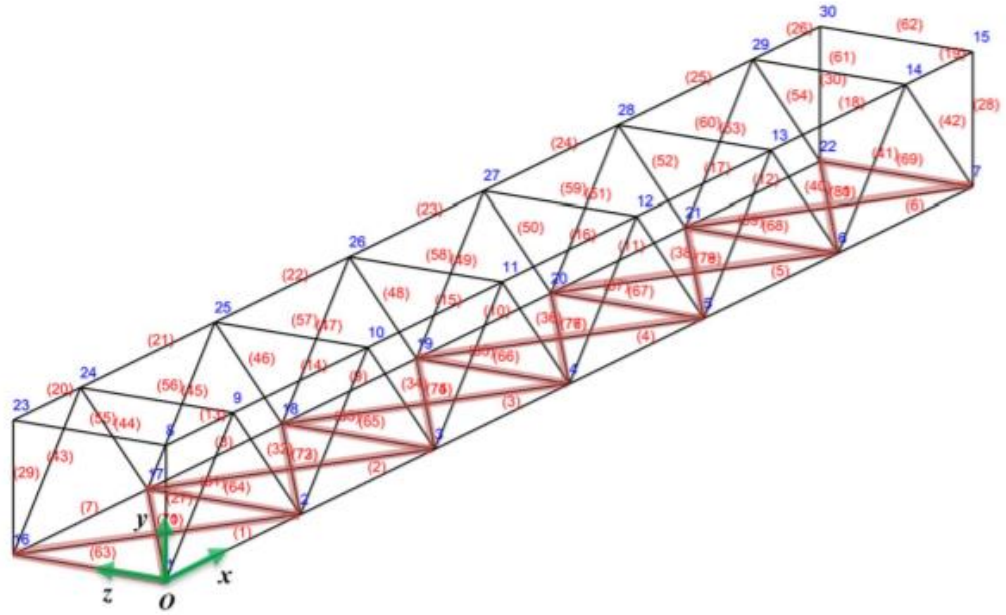


Fig. 6. FE model of the pedestrian bridge with node and element numbers (node numbers: numbers without parentheses; and element numbers: numbers with parentheses).

Case study

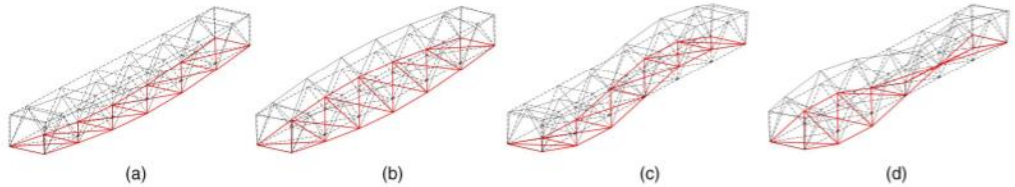


Fig. 7. Natural frequencies and mode shapes calculated from the initial FE model: (a) Mode 1 (5.9609 Hz); (b) Mode 2 (9.3599 Hz); (c) Mode 3 (15.4479 Hz); and (d) Mode 4 (20.3188 Hz).

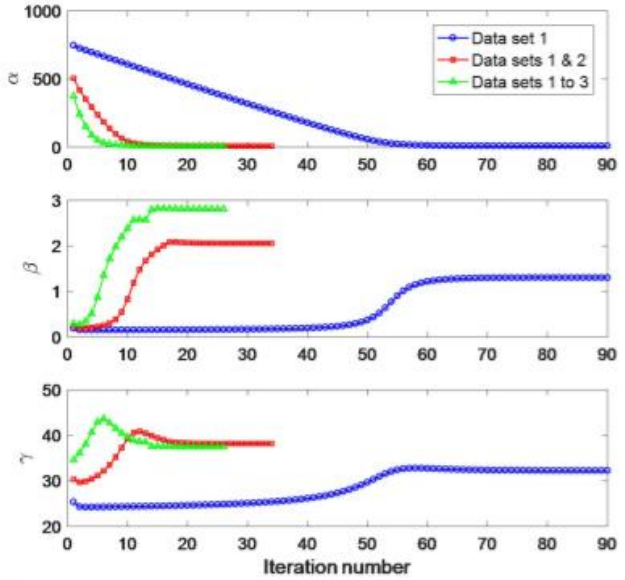


Fig. 9. Convergence history of three hyperparameters with different numbers of data sets.

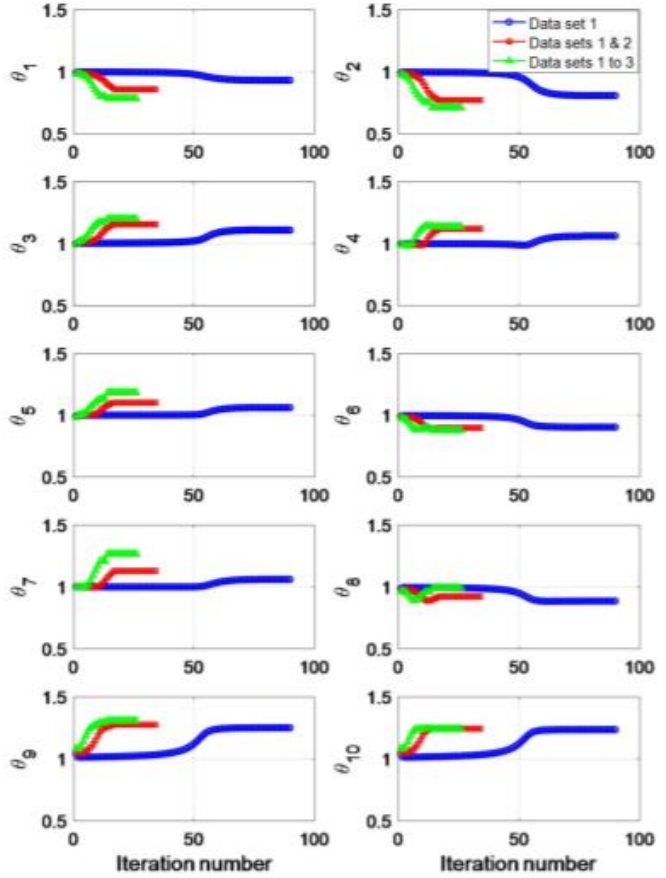


Fig. 8. Convergence history of FE model parameters with different numbers of data sets.

- The adjustment of model parameters gradually increases as the iteration proceeds before the convergence is achieved

Case study

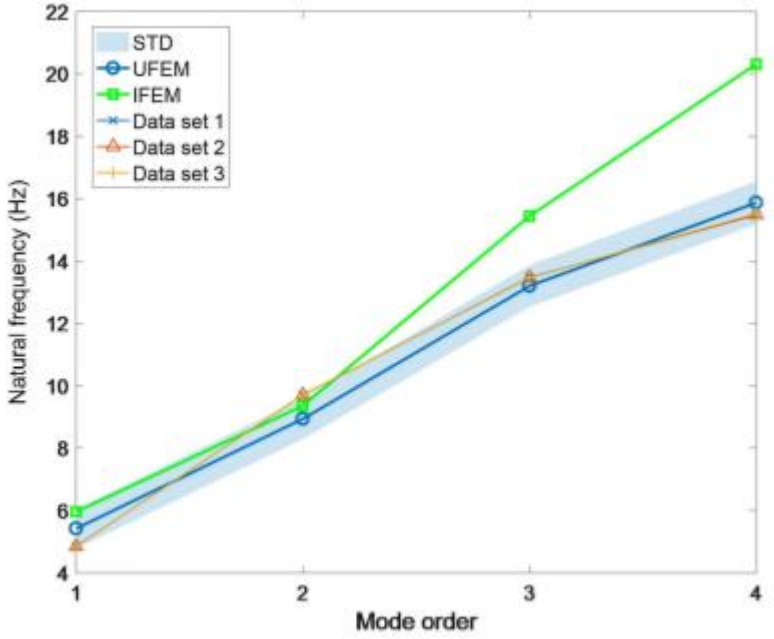


Fig. 10. Comparison of experimental natural frequencies with model-predicted results.

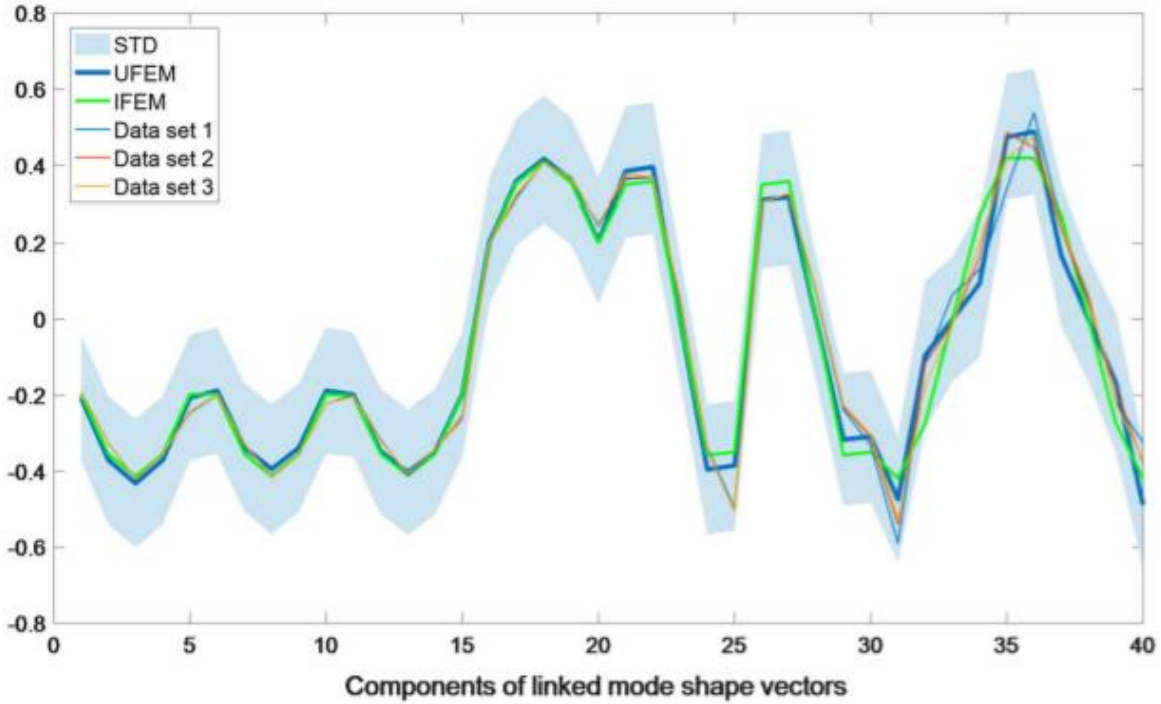


Fig. 11. Comparison of experimental mode shapes with model-predicted results.

- The comparisons of experimental natural frequencies and mode shapes with those predicted from the updated FE model

Conclusions

- In this paper, a practical statistical framework for structural model updating and prediction based on the Bayesian regularization using incomplete modal data is proposed

- One of the significant features of the proposed framework lies in that the existing nonlinear least-squares algorithm, specifically the trust-region-reflective algorithm, is fully explored to solve the optimization problem of Bayesian inference of regularization hyperparameters

- Based on the structural FE model calibrated with measured modal data, the posterior predictive distribution over natural frequencies and mode shapes are obtained

- A reasonable quantitative evaluation of the prediction uncertainty can be provided to give confidence to the updated model

- One can obtain a calibrated FE model that is as consistent as possible with the measured modal data with a relatively modest amount of model parameter correction so as to improve the noise robustness of dating procedure

☆Thank you for listening ☆

Learning Continuous Image Representation with Local Implicit Image Function (CVPR 2021 oral)

패턴인식 발표 November 20th, 2021

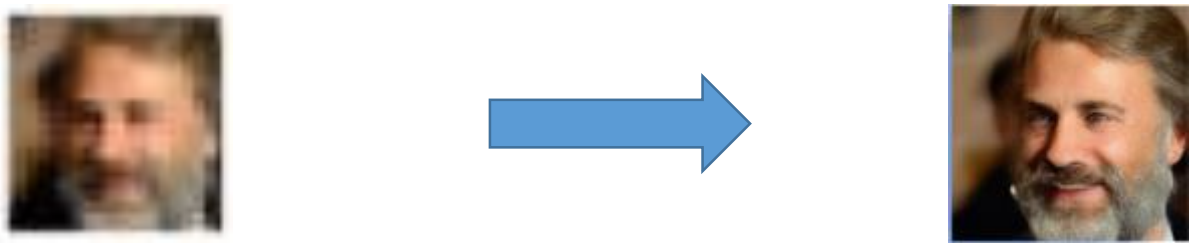
Presenter : 홍상민

Computer Vision Lab.
Seoul National University

Outline

- What is Super Resolution
- Learning Implicit Function Space
- Local Implicit Image Function
- Motivation/Contribution
- Implicit Neural Representation
- Overall Pipeline
- Conclusion

What is Super Resolution

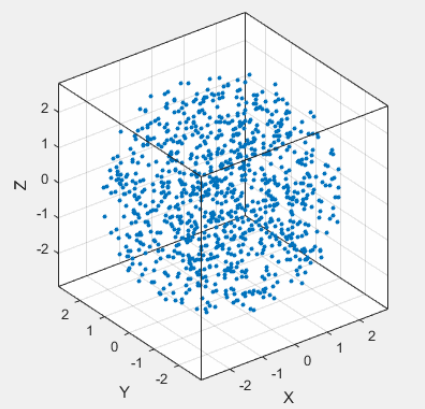


- Super-resolution(SR) is a technique that aims to enhance the resolution of an image by adding missing information.
- From low resolution images (LR), reconstruct high resolution images (HR)

Implicit Neural Representation

- **Implicit neural representation**: Constructed with MLPs which maps coordinates to signal. Ex) 3D space

$$x^2 + f(x)^2 - 1 = 0$$



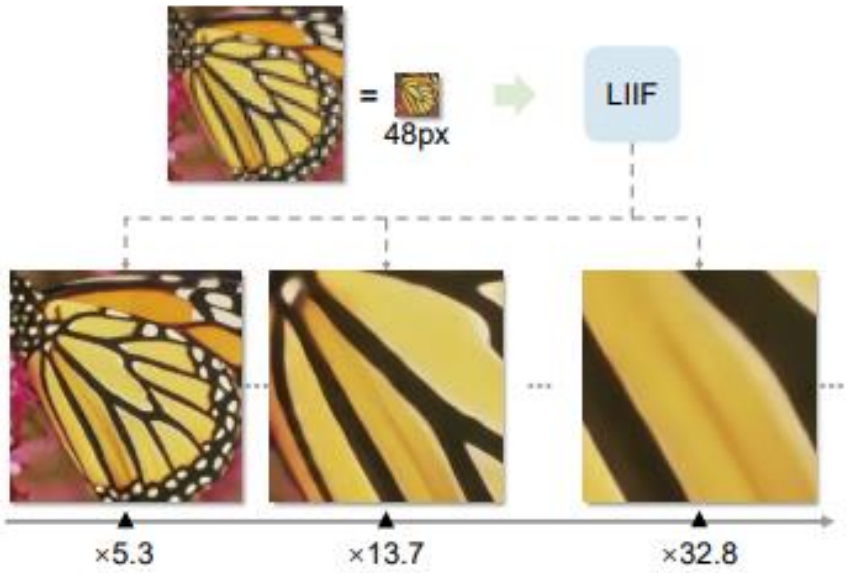
if (x, y, z) **on surface** $\longrightarrow$ $f(x, y, z) = 0$

if (x, y, z) **inside surface** $\longrightarrow$ $f(x, y, z) > 0$

if (x, y, z) **ouside surface** $\longrightarrow$ $f(x, y, z) < 0$

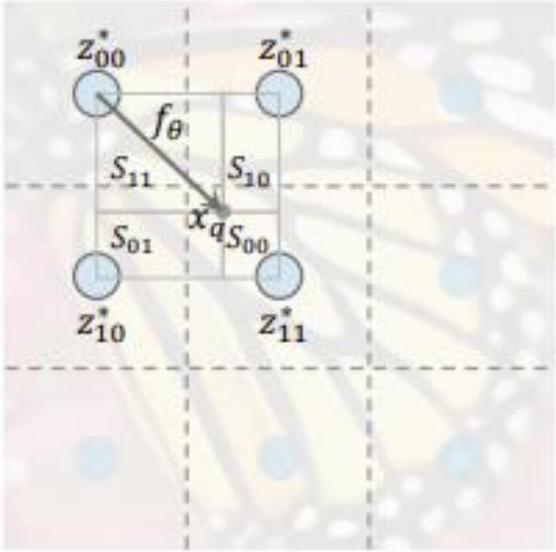
Motivation/Contribution

- What about applying the implicit neural representation in images?
- Novel method for representing image continuously
- Great Scalability: allows extrapolation to even x30



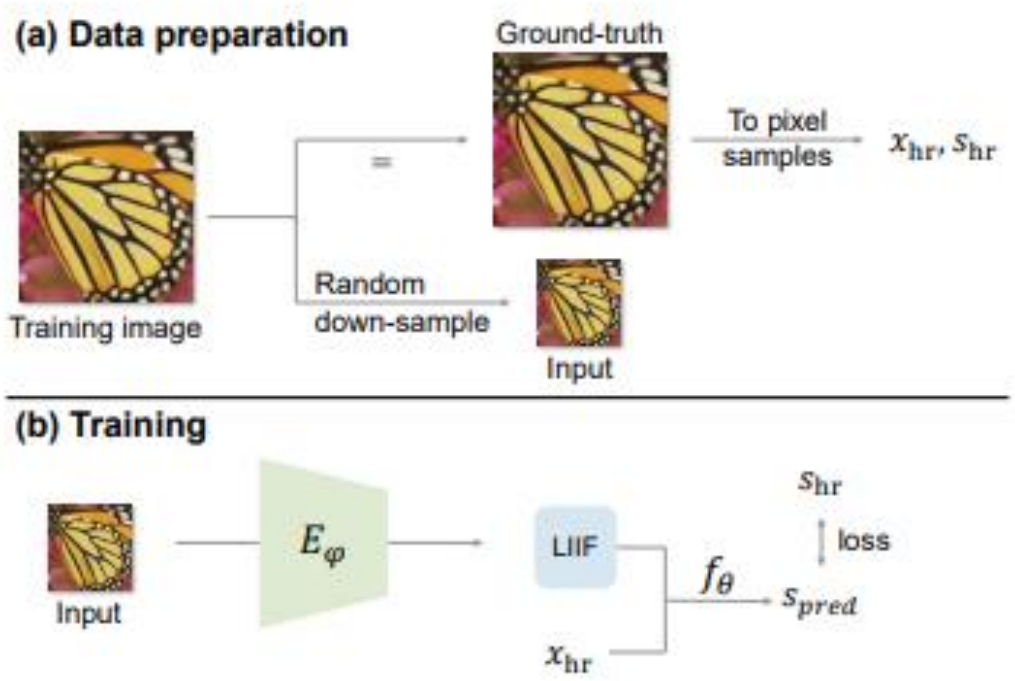
Local Implicit Image function

- **$s = f_\theta(z, x)$**
- z -> nearest latent vector from the image
- x -> coordinate of the image
- s-> predicted signal (RGB value)



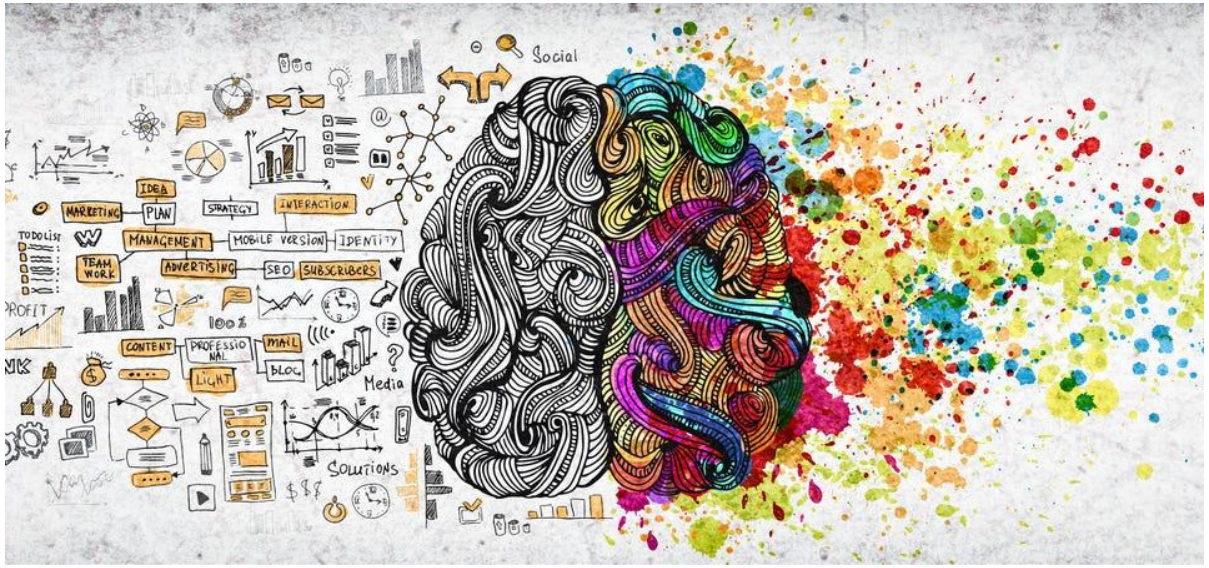
Overall Pipeline

- **Data preparation**: Random downsample the training image to get the input.
- It is self-supervised method
- Train encoder(obtain 2D feature map) together with LIIF
- $\mathbf{x}_{hr}$ is the coordinate that is used for query on LIIF



Conclusion

- Great work that uses implicit representation for new application
- Our world is continuous and there is lots of areas that implicit function can represent
- Use your creativity and use this paper as inspiration to your work.



Normalizing Kalman Filters for Multivariate Time Series Analysis

Emmanuel de Bézenac et al., NIPS 2020

Junghyeon Kwon

WCISL Wireless Communication and Information Systems Laboratory

Normalizing Kalman Filters

- **Generative model**
  - The modelling of non-Gaussian multivariate time series data with nonlinear inter-dependencies that has Kalman-like recursive updates for filtering:

$$
\begin{aligned}
& & \mathbf{l}_1 &\sim \mathcal{N}(\mu_1, \Sigma_1) & & && \text{(initial state)} \\
&\text{(NKF model)} & \mathbf{l}_t &= F_t \mathbf{l}_{t-1} + \boldsymbol{\epsilon}_t, & \boldsymbol{\epsilon}_t &\sim \mathcal{N}(0, \Sigma_t), && \text{(transition dynamics)} \\
& & \mathbf{y}_t &= f_t(A_t^T \mathbf{l}_t + \boldsymbol{\varepsilon}_t), & \boldsymbol{\varepsilon}_t &\sim \mathcal{N}(0, \Gamma_t). && \text{(observation model)}
\end{aligned}
$$

  - l_t: latent state, F_t: transition matrix, A_t: emission matrix, $\mathbf{y}_t$: observation, with Λ and $\Theta = (\mu_1, \Sigma_1, \{\Gamma_t, A_t\}_{t\geq 1}, \{\Sigma_t, F_t\}_{t\geq 2})$.
  - We consider a flexible nonlinear transformation for the observation model, assuming invertibility of f_t.

Normalizing Kalman Filters

- **Generative model**
  - The probability density of an observation:

$$p(y_t|l_t;\Theta,\Lambda) = p_z(f_t^{-1}(y_t)|l_t;\Theta)\left|\det\left[\mathrm{Jac}_{y_t}(f_t^{-1})\right]\right|$$

  - Computing the density raises several issues:
    ① Finding a flexible while ensuring invertibility.
    ② Being able to compute the inverse efficiently.
    ③ Tractability of the computation of the Jacobian term when the number of time series N is large.

Normalizing Kalman Filters

- **Normalizing flows**
  - Taking inspiration from normalizing flows, which are invertible neural networks that typically transform isotropic Gaussians to fit a more complex data distribution.
  - In this approach, we apply these invertible neural networks to temporal data, using them to map the distribution p_z given by the LGM to the complex data distribution.
  - *Inference and learning: The filtered distribution $p(l_t|y_{1:t};\Theta,\Lambda)$ is essential as it determines our current belief on the state having observed all the data up to time t.*
  - *Despite the nonlinear nature of f_t, the filtered distribution $p(l_t|y_{1:t};\Theta,\Lambda)$ remains Gaussian and its parameters can be computed in closed form similarly to the Kalman Filter.*

Normalizing Kalman Filters

- **Proposition 1**
  - The filtered distributions of NKF model are Gaussian and are given by the filtered distributions of the corresponding LGM with pseudo-observations $z_t \coloneqq f_t^{-1}(y_t), t \geq 1$. That is, $p(l_t|y_{1:t}; \Theta, \Lambda) = p_{LGM}(l_t|z_{1:t}; \Theta)$ where p_{LGM} refers to the distribution given by the LGM.

- **Proposition 2**
  - The likelihood of the parameters (Θ, Λ) of the NKF model given the observations $\{y_{1:T}\}$ can be computed as

$$\ell(\Theta, \Lambda) = p(y_{1:T}; \Theta, \Lambda) = \prod_{t=1}^{T} p_{\mathrm{LGM}}(z_t|z_{1:t-1}; \Theta) \left|\det\left[\mathrm{Jac}_{z_t}(f_t)\right]\right|^{-1}$$

  where $z_t \coloneqq f_t^{-1}(y_t)$ and $p_{LGM}(z_t|z_{1:t-1}; \Theta)$ denotes the predictive distribution of LGM.

Normalizing Kalman Filters

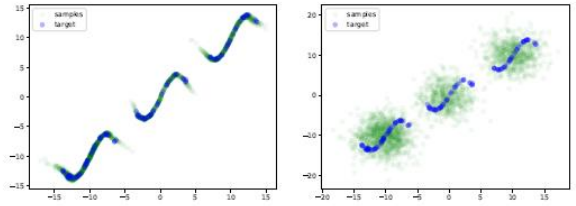
- **Application: Forecasting and missing values**

$$
\begin{aligned}
F_t, A_t, \Sigma_t, \Gamma_t &= \sigma(h_t; \Phi), & h_t &= \Psi(x_t, h_{t-1}; \Phi), \\
l_t &= F_t l_{t-1} + \epsilon_t, & \epsilon_t &\text{ sampled from } \mathcal{N}(0, \Sigma_t), \\
y_t &= f_t(A_t^T l_t + \varepsilon_t), & \varepsilon_t &\text{ sampled from } \mathcal{N}(0, \Gamma_t), \qquad t = T+1, \ldots, T+\tau.
\end{aligned}
$$

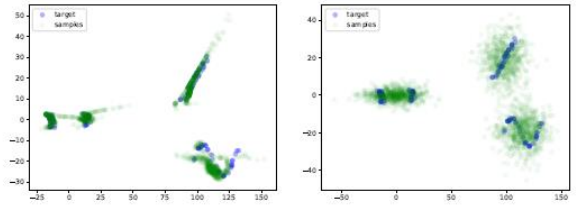
- In contrast to alternative deep learning approaches, this generative procedure is not autoregressive in the sense that observations are not fed to the model.
- When missing for t, we can compute the filtered distribution $p(l_{t-1}|y_{1:t-1}; \Theta, \Lambda)$ and then start the prediction step starting from the filtered distribution at time $t-1$.

Experiments

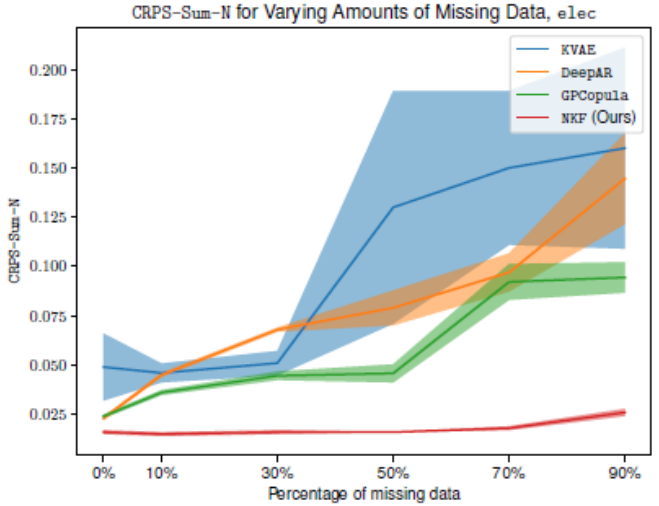
- **Qualitative results and missing data experiment**



(a) S1 : Results with (left) and without (right) NF.



(b) S2 : Results with (left) and without (right) NF.



- The variant captures the daily modes correctly, but assumptions such as Gaussianity and independence between time series introduce errors.
- We observe that not only does NKF outperform other approaches aby a large margin, but its error also increases slower than in other methods when the percentage of missing data is increased.

Thank you

Learning Continuous Image Representation with Local Implicit Image Function (CVPR 2021)

Yinbo Chen, Sifei Liu, Xiaolong Wang

Junhyeong Kwon

Link: https://openaccess.thecvf.com/content/CVPR2021/html/Chen_Learning_Continuous_Image_Representation_With_Local_Implicit_Image_Function_CVPR_2021_paper.html

Introduction

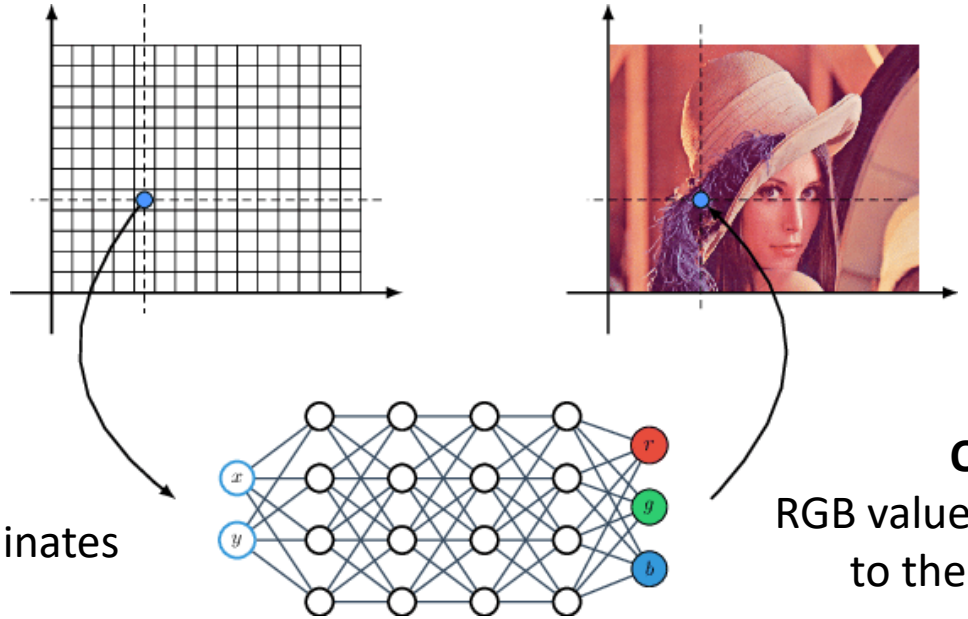
- Super-Resolution

Input
(Low resolution image)

Output
(Super-resolved high resolution image)

Introduction

- Implicit Neural Representation

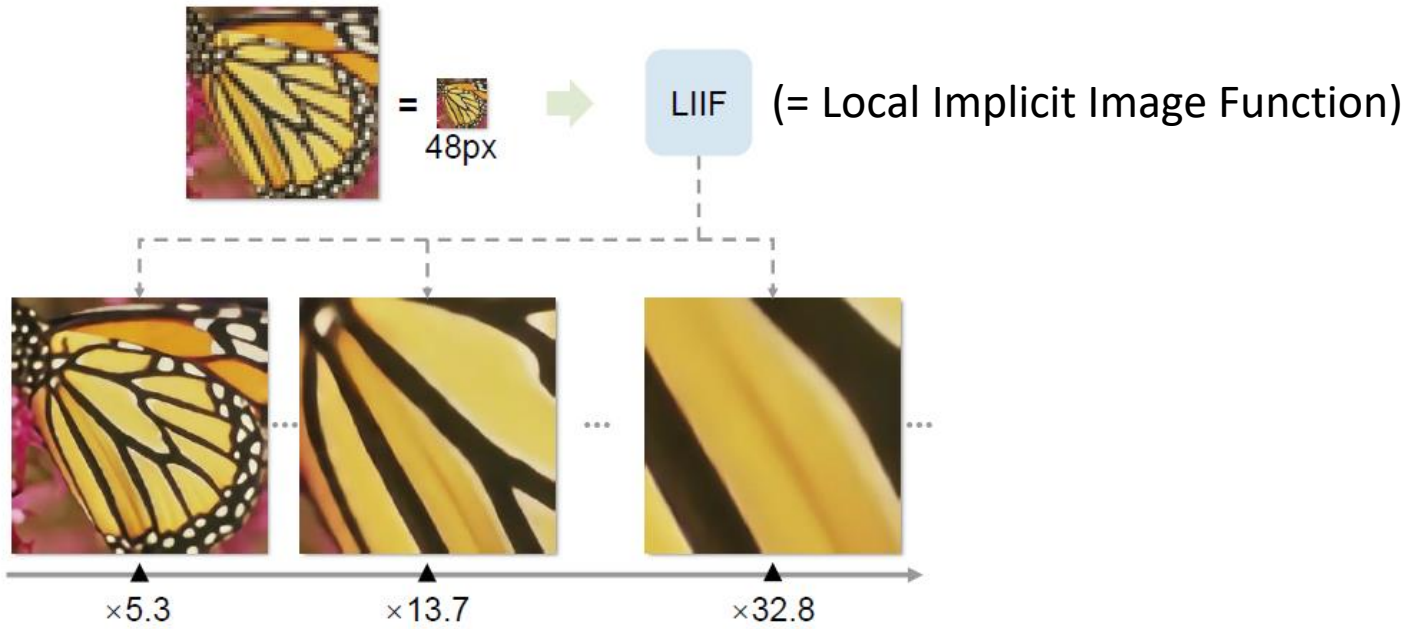


Input:
image coordinates

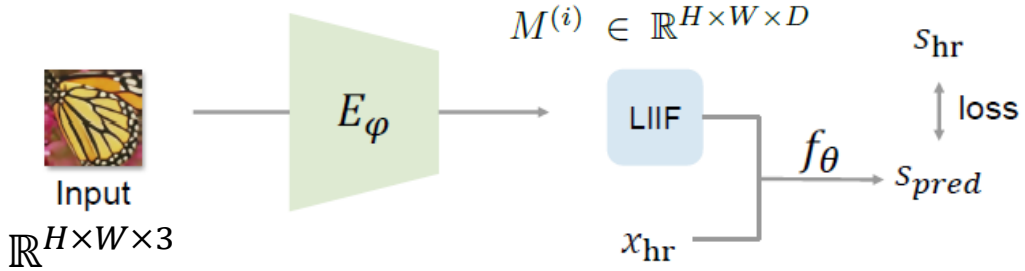
Output:
RGB values corresponding
to the coordinates

Contributions

- By modeling an image as a function defined in a **continuous domain**, the image can be restored and generated **in arbitrary resolution**.



Overall Architecture



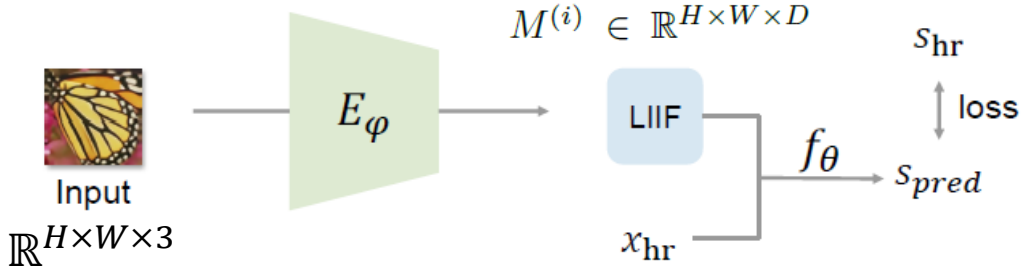
E_φ: Encoder parameterized with φ

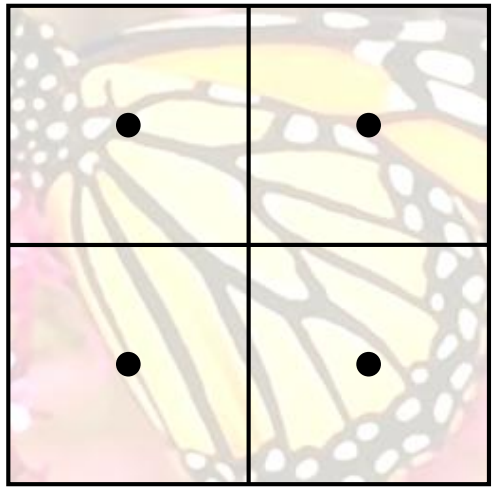
f_θ: Decoding function parameterized with θ

x_{hr}: center coordinates of pixels in the HR image domain

s_{hr}: corresponding RGB values of the pixels

Example: **2x SR task**





LIIF representation
$M^{(i)} \in \mathbb{R}^{2\times 2\times D}$

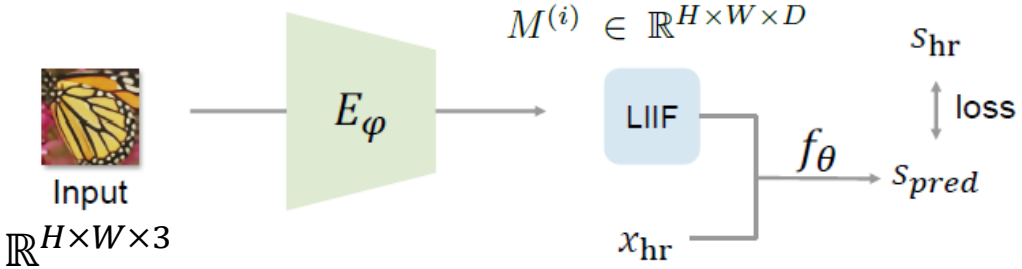
E_φ: Encoder parameterized with φ

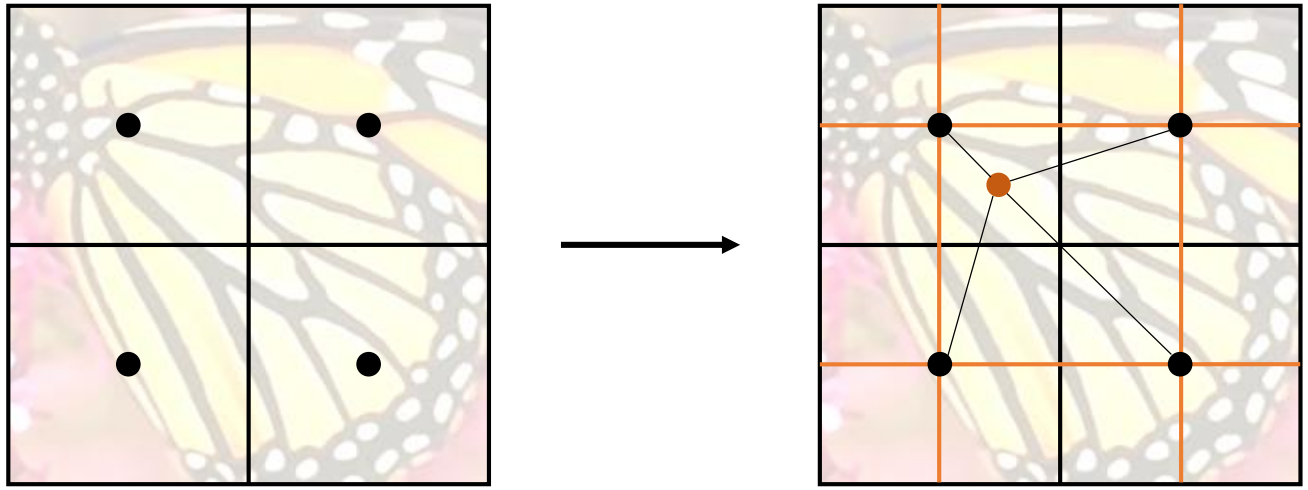
f_θ: Decoding function parameterized with θ

x_{hr}: center coordinates of pixels in the HR image domain

s_{hr}: corresponding RGB values of the pixels

Example: **2x SR task**





LIIF representation
$M^{(i)} \in \mathbb{R}^{2\times 2\times D}$

● : x_{hr}

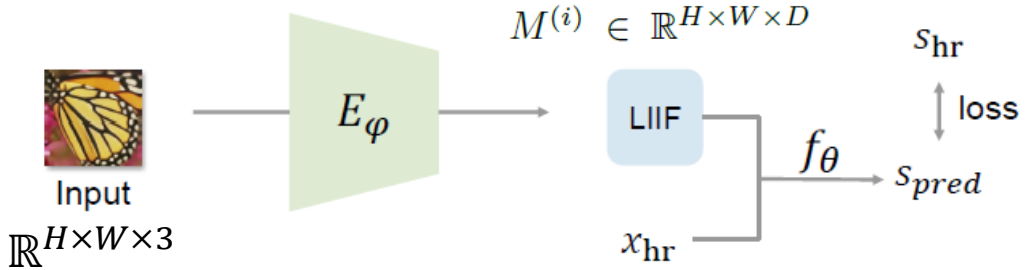
E_φ: Encoder parameterized with φ

f_θ: Decoding function parameterized with θ

x_{hr}: center coordinates of pixels in the HR image domain

s_{hr}: corresponding RGB values of the pixels

Example: **2x SR task**

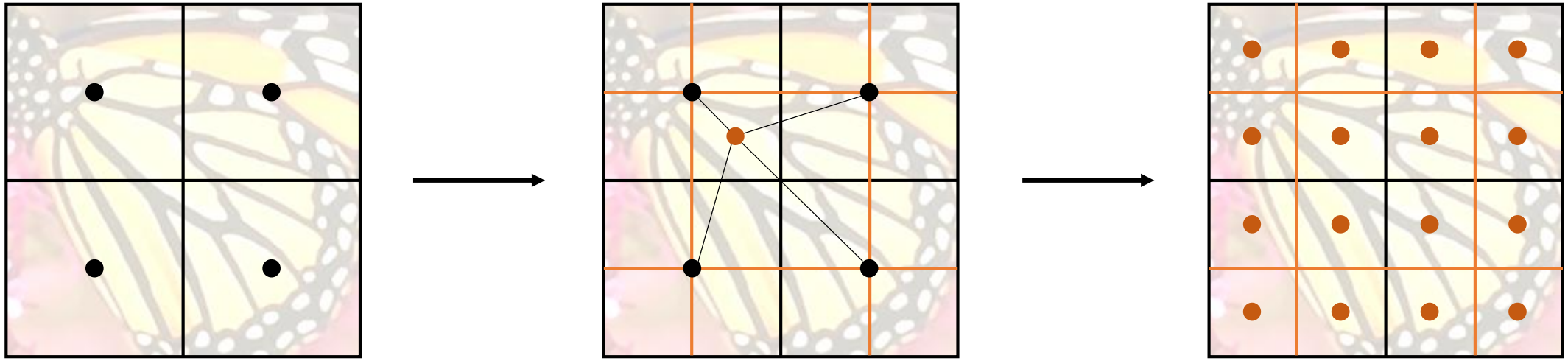


E_φ: Encoder parameterized with φ

f_θ: Decoding function parameterized with θ

x_{hr}: center coordinates of pixels in the HR image domain

s_{hr}: corresponding RGB values of the pixels



LIIF representation
$M^{(i)} \in \mathbb{R}^{2\times 2\times D}$

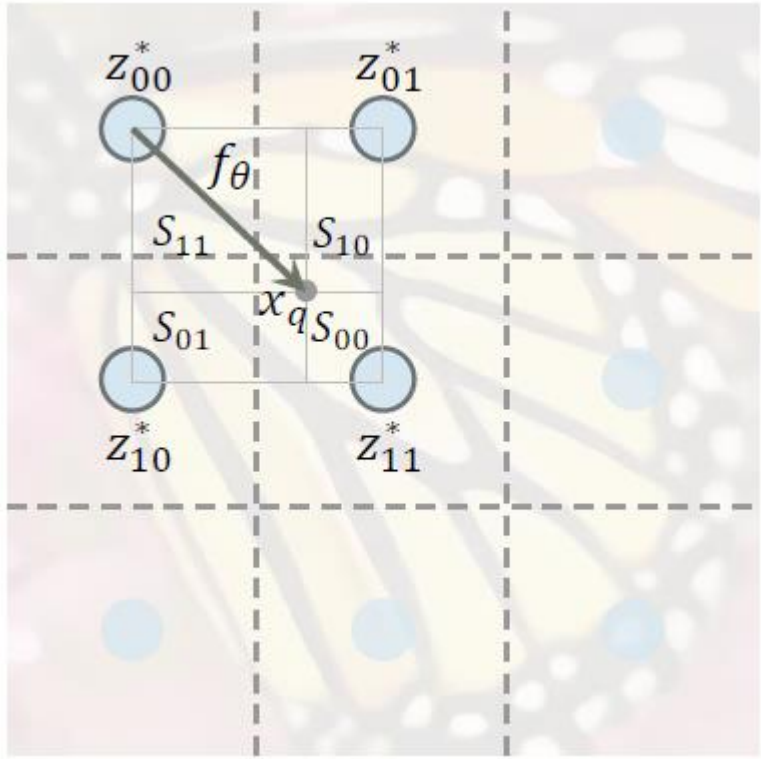
$s_{pred} \in \mathbb{R}^{4\times 4\times 3}$

Local Implicit Image Function (LIIF)

- Local ensemble

$$I^{(i)}(x_q) = \sum_{t \in \{00,01,10,11\}} \frac{S_t}{S} \cdot f_\theta(z_t^*, x_q - v_t^*)$$

x_q: query coordinate
z^*: nearest feature vector
v^*: z^*'s coordinate
f_θ: neural implicit function



Experiments

Method	In-distribution			Out-of-distribution				
	×2	×3	×4	×6	×12	×18	×24	×30
Bicubic [24]	31.01	28.22	26.66	24.82	22.27	21.00	20.19	19.59
EDSR-baseline [24]	34.55	30.90	28.94	-	-	-	-	-
EDSR-baseline-MetaSR♯ [15]	34.64	30.93	28.92	26.61	23.55	22.03	21.06	20.37
EDSR-baseline-LIIF (ours)	34.67	30.96	**29.00**	**26.75**	**23.71**	**22.17**	**21.18**	**20.48**
RDN-MetaSR♯ [15]	35.00	31.27	29.25	26.88	23.73	22.18	21.17	20.47
RDN-LIIF (ours)	34.99	31.26	29.27	**26.99**	**23.89**	**22.34**	**21.31**	**20.59**

Table 1: **Quantitative comparison on DIV2K validation set (PSNR (dB)).** ♯ indicates ours implementation. The results that surpass others by 0.05 are bolded. EDSR-baseline trains different models for different scales. MetaSR and LIIF use one model for all scales, and are trained with continuous random scales uniformly sampled in ×1–×4.

Experiments

Dataset	Method	In-distribution			Out-of-distribution	
		×2	×3	×4	×6	×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**

Table 2: **Quantitative comparison on benchmark datasets (PSNR (dB)).** ♯ indicates ours implementation. The results that surpass others by 0.05 are bolded. RDN trains different models for different scales. MetaSR and LIIF use one model for all scales, and are trained with continuous random scales uniformly sampled in ×1–×4.

Qualitative results: **https://yinboc.github.io/liif/**

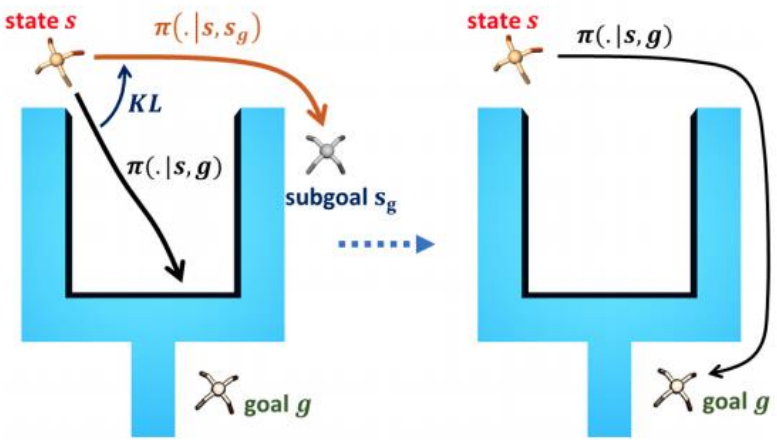
Thank you!

Goal-conditioned Reinforcement Learning with Imagined Subgoals

Elliot Chane-Sane, Cordelia Schmid, Ivan Laptev

International Conference on Machine Learning (ICML) 2021

Overview



Goal-conditioned RL problem

(Ant navigation, Vision-based robotic manipulation)

Two policies

- High-level policy π_H : predicting imagined subgoals
- Target policy π : goal-reaching control policy

Learning the goal-reaching policy $\pi(\cdot|s, g)$ with the subgoal-reaching policy $\pi(\cdot|s, s_g)$ as a guidance.

Goal-Conditioned Actor-Critic

states $s \in \mathcal{S}$
goals $g \in \mathcal{G}$
actions $a \in \mathcal{A}$

Expected discounted return

$$J(\pi) = \mathbb{E}_{g \sim \rho_g, \tau \sim d^\pi(.|g)}[\sum_t \gamma^t r(s_t, a_t, g)]$$

Distribution conditioned on given goal

$$d^\pi(\tau|g) = \rho_0(s_0) \prod_t \pi(a_t|s_t, g) p(s_{t+1}|s_t, a_t)$$

Assume $\mathcal{S} = \mathcal{G}$: states and goals co-exist in the same space.

Reward r = -1 for every step until reach the goal.

High-Level Policy

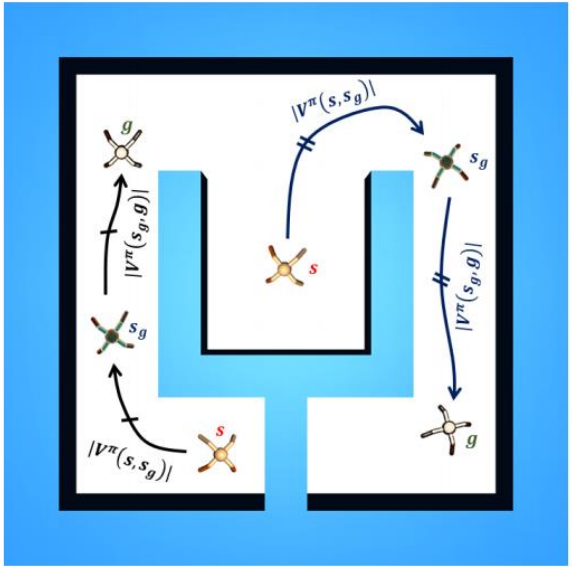
Predicts imagined subgoals s_g conditioned on s and g.

Value function: $V^\pi(s, g)$

Since r = -1, $|V^\pi(s^i, s^j)|$ becomes the measure of the distance between s^i and s^j.

minimize $C_\pi(s_g|s, g) = \max\left(|V^\pi(s, s_g)|, |V^\pi(s_g, g)|\right)$

$$\pi^H_{k+1} = \arg\min_{\pi^H} \mathbb{E}_{(s,g)\sim D, s_g\sim\pi^H(.|s,g)}[C_\pi(s_g|s, g)]$$



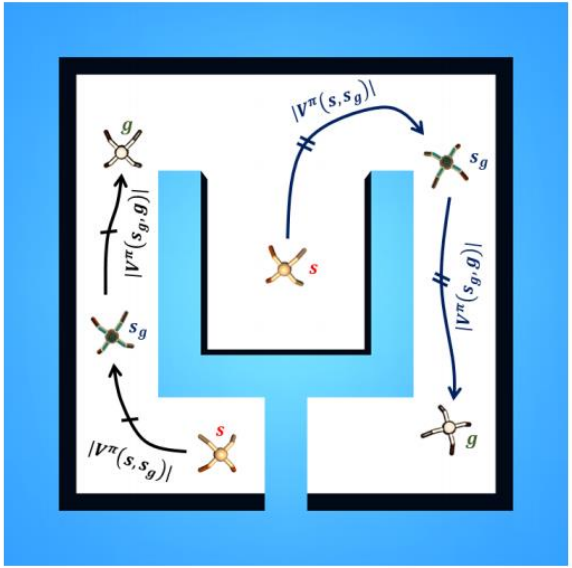
High-Level Policy

$$\pi^H_{k+1} = \arg\min_{\pi^H} \mathbb{E}_{(s,g)\sim D, s_g \sim \pi^H(.|s,g)} [C_\pi(s_g|s,g)]$$

$$\pi^H_{k+1} = \arg\max_{\pi^H} \mathbb{E}_{(s,g)\sim D, s_g \sim \pi^H(.|s,g)} \left[A^{\pi^H_k}(s_g|s,g) \right]$$
$$\text{s.t. } D_{\mathrm{KL}}\left(\pi^H(.|s,g) \,||\, p_s(.)\right) \leq \epsilon,$$

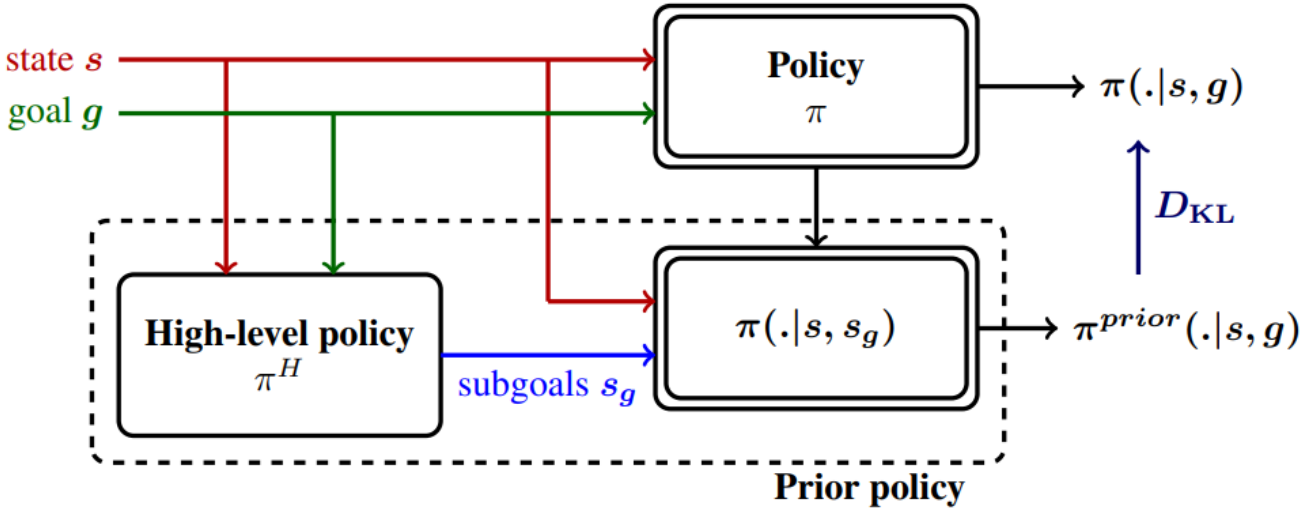
where advantage function

$$A^{\pi^H_k}(s_g|s,g) = \mathbb{E}_{\hat{s_g} \sim \pi^H_k(.|s,g)} [C_\pi(\hat{s_g}|s,g)] - C_\pi(s_g|s,g)$$



Target Policy Improvement

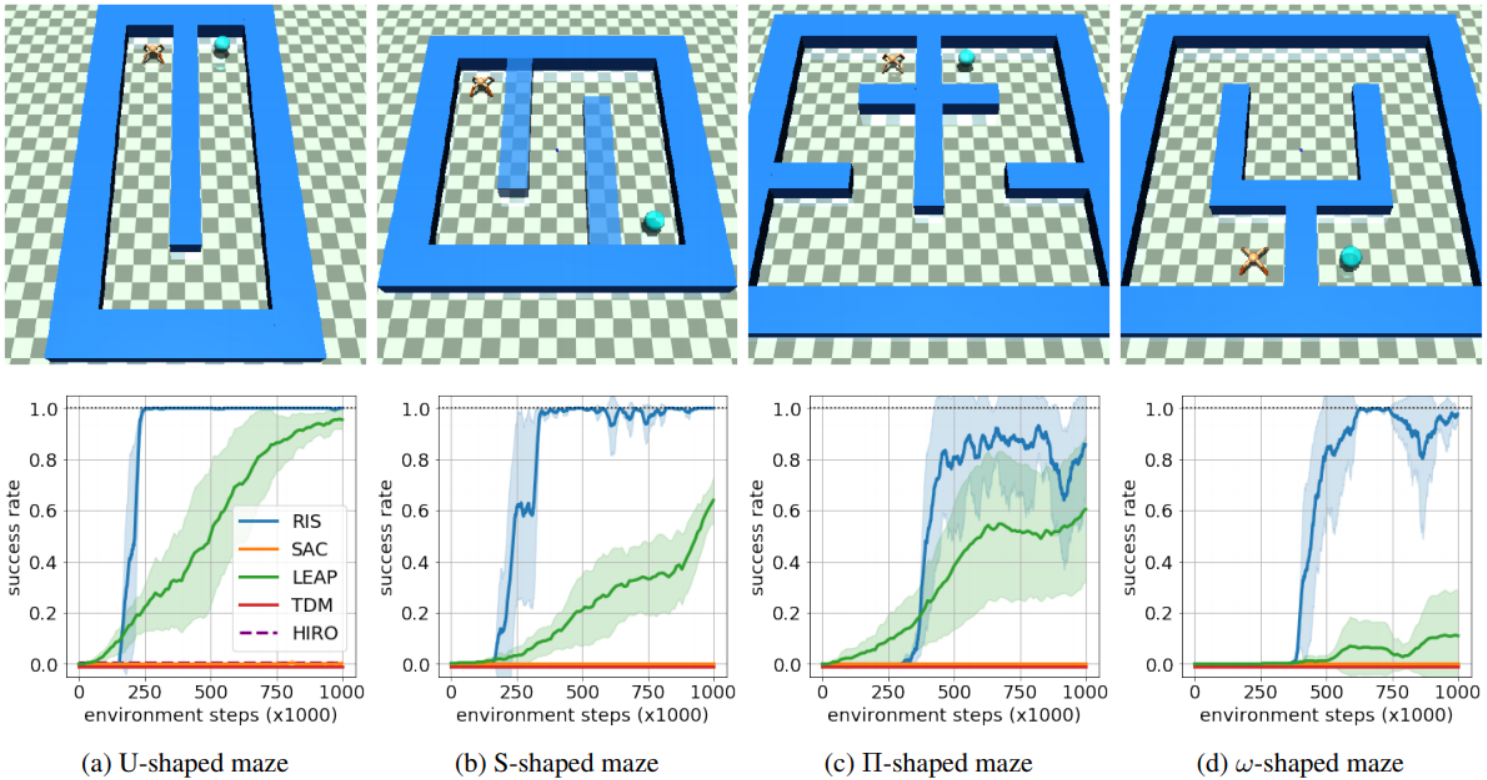
KL constraint: $D_{\mathrm{KL}}\left(\pi(.|s,g) \,||\, \pi(.|s,s_g)\right) \leq \epsilon.$



$$\pi_{\theta_{k+1}} = \arg\max_{\theta} \mathbb{E}_{(s,g)\sim D} \mathbb{E}_{a\sim\pi_\theta(.|s,g)}$$

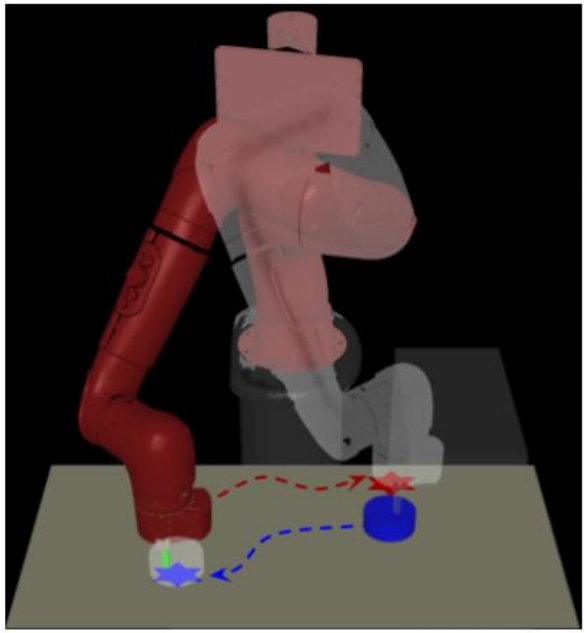
$$\left[Q^{\pi}(s,a,g) - \alpha D_{\mathrm{KL}}\left(\pi_\theta(.|s,g) \,||\, \pi_k^{prior}(.|s,g)\right)\right]$$

Experiments - Ant Navigation

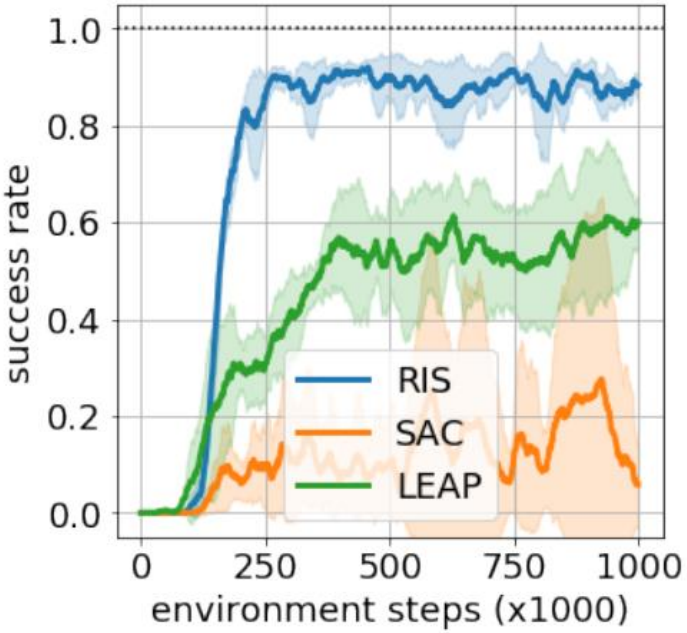


(a) U-shaped maze (b) S-shaped maze (c) Π-shaped maze (d) ω-shaped maze

Experiments - Robotic Manipulation



(a) Illustration of the robotic manipulation task



(b) Comparison to prior works

Deep repulsive clustering of ordered data based on order-identity decomposition

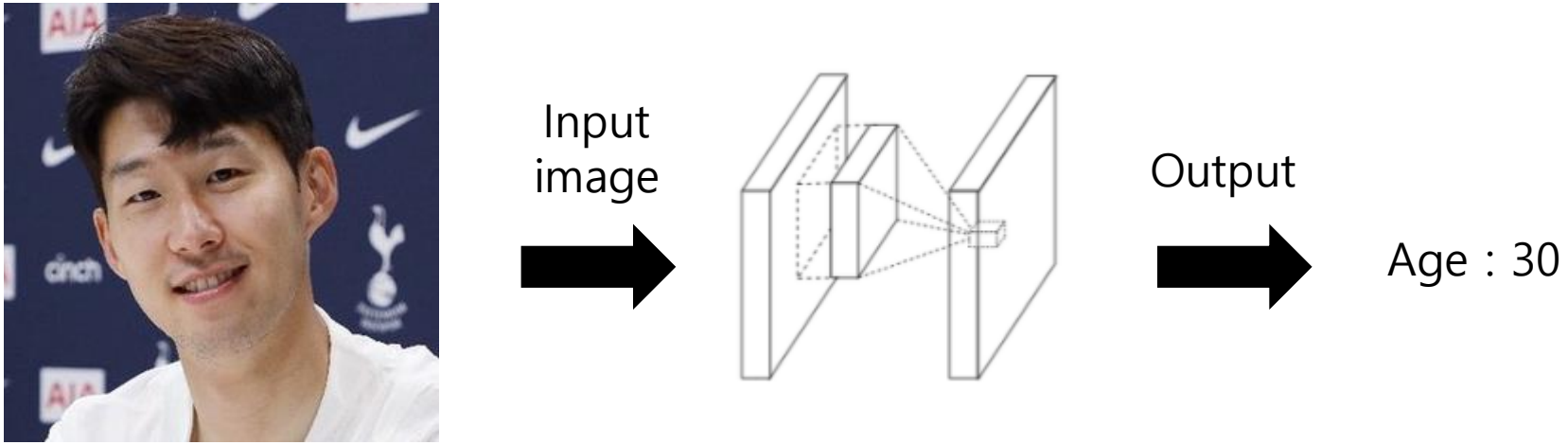
International Conference on Learning Representations (ICLR). 2020.

Radio Technology Lab. SNU

Keunwoo Kim

1. Motivation

- Age estimation
  - Estimates someone's age from input image



1. Motivation

- Age estimation
  - Trains model with labeled data (label : age)

Age

30

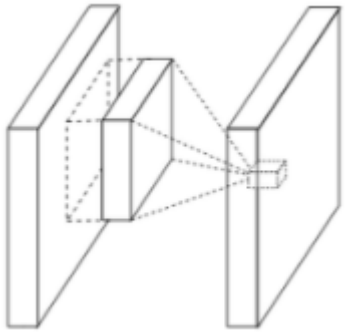
Age

24

Age

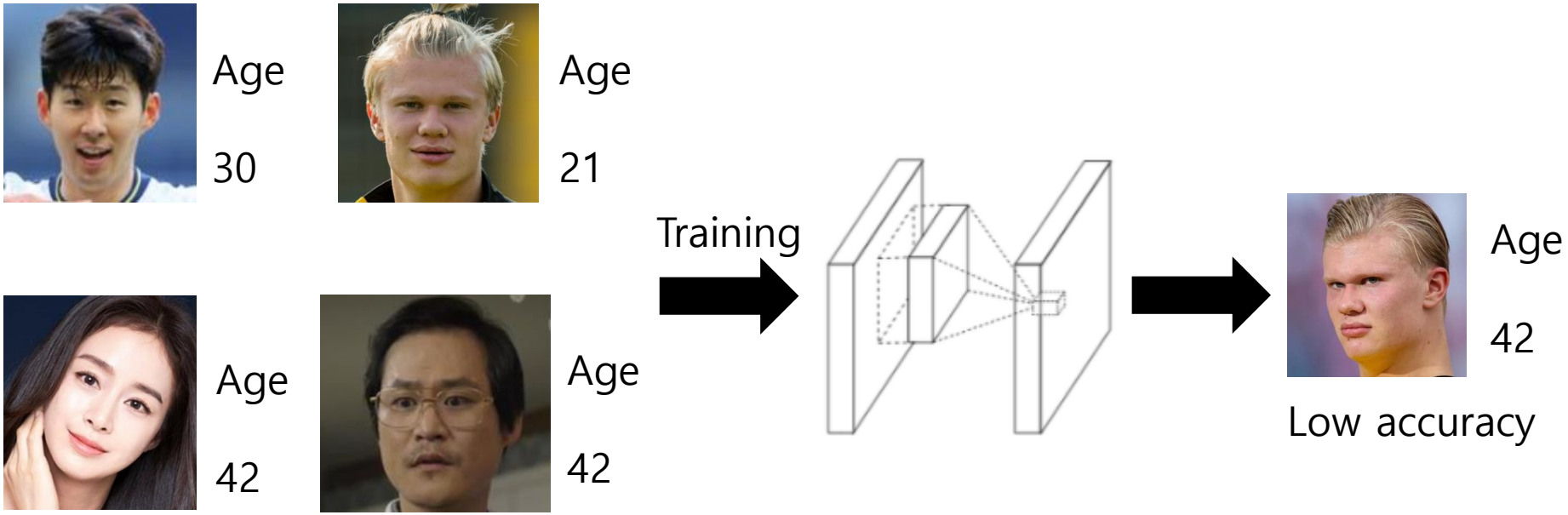
21

Training



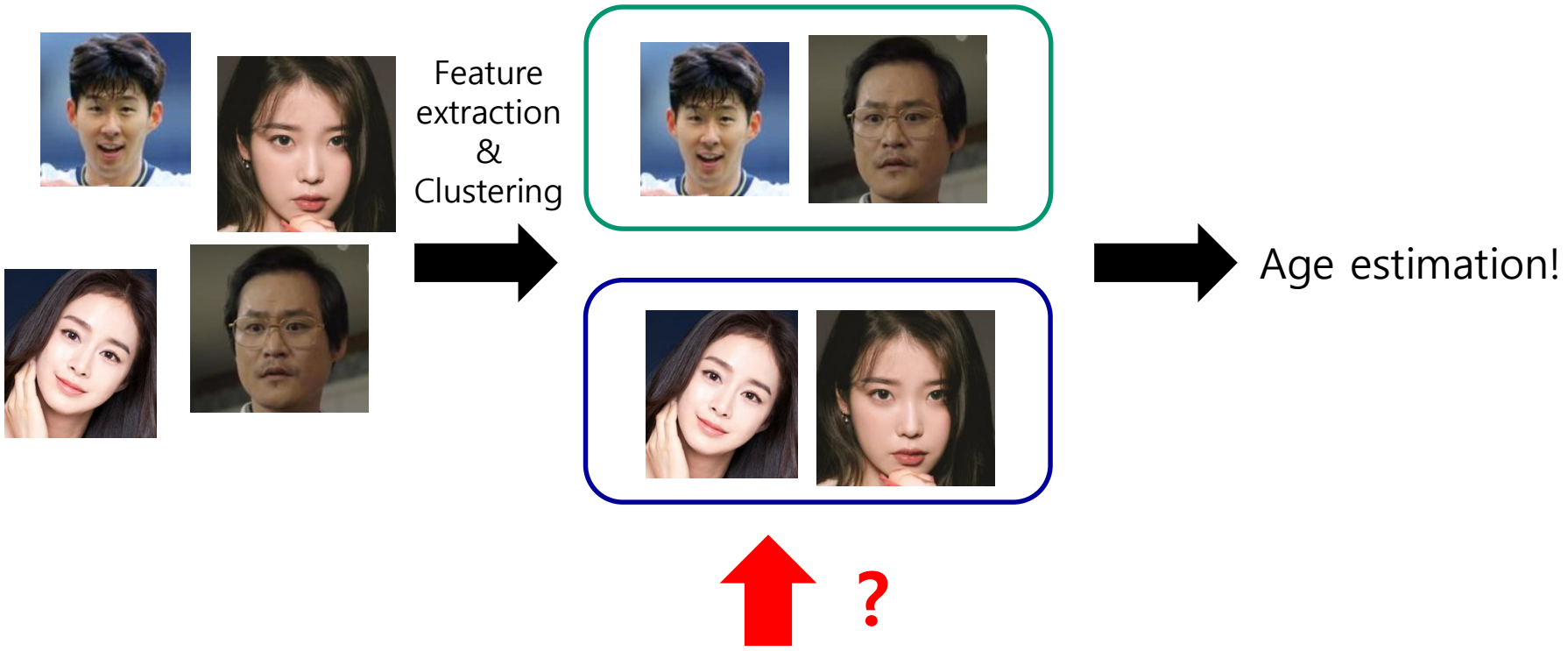
1. Motivation

- Age estimation
  - Different race, gender, etc. cause low accuracy of estimated age



1. Motivation

- Age estimation
  - Divides data into clusters & estimates its age!



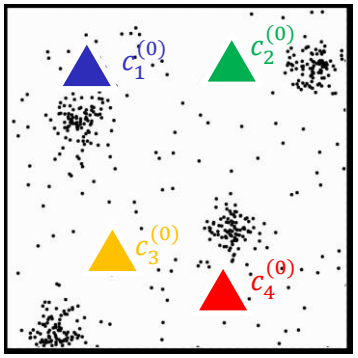
2. Proposed algorithm

- Unsupervised learning
  - k-means clustering algorithm
    - For simplicity, let $d_{id} = 2,\ k = 4,\ j = 1,2,\dots,k$

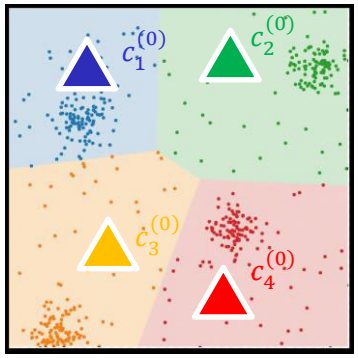
Input x

Feature $h_{id}^{x} \in \mathbb{R}^{d_{id}}$

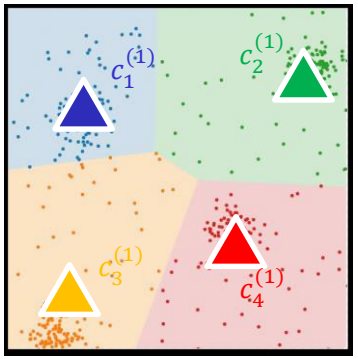
$h_{id}^{x\ T} h_{id}^{x} = 1$



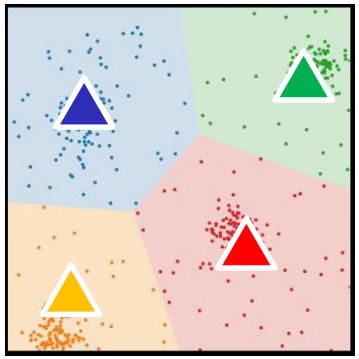
1. $c_j^{(0)} \rightarrow$ Random



2. $C_j^{(t)} = \{x: \min(\left|h_{id}^{x} - c_j^{(t)}\right|^2)\}$

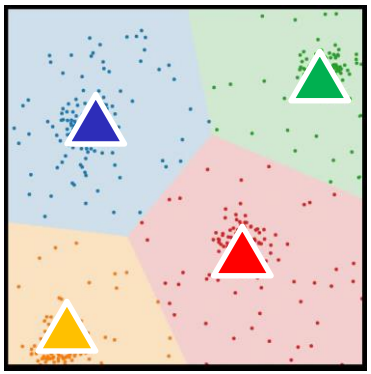


3. $c_j^{(t+1)} = \frac{1}{|C_j^{(t)}|} \sum_{x \in C_j^{(t)}} h_{id}^{x}$



4. Repeat 2~3

Until converge



k clusters

2. Proposed algorithm

- Unsupervised learning
  - Proposed algorithm (Deep repulsive clustering (DRC))
    - Add repulsive term when choosing centroid c_j

Spherical k-means clustering Repulsive term

Cost function $J\left(\{C_j\}_{j=1}^{k}, \{c_j\}_{j=1}^{k}\right) = \sum_{j=1}^{k}\sum_{x\in C_j}((h_{id}^{x})^T c_j - \alpha\frac{1}{k-1}\sum_{l\neq j}(h_{id}^{x})^T c_l)$

Input x

Feature $h_{id}^{x} \in \mathbb{R}^{d_{id}}$

$h_{id}^{x\,T} h_{id}^{x} = 1$

1. Centroid c_j rule

$$\max J\left(\{c_j\}_{j=1}^{k}\right)$$
$$s.t.\, c_j^T c_j = 1 \;\; (j = 1,2,\dots,k)$$

2. Nearest neighbor rule

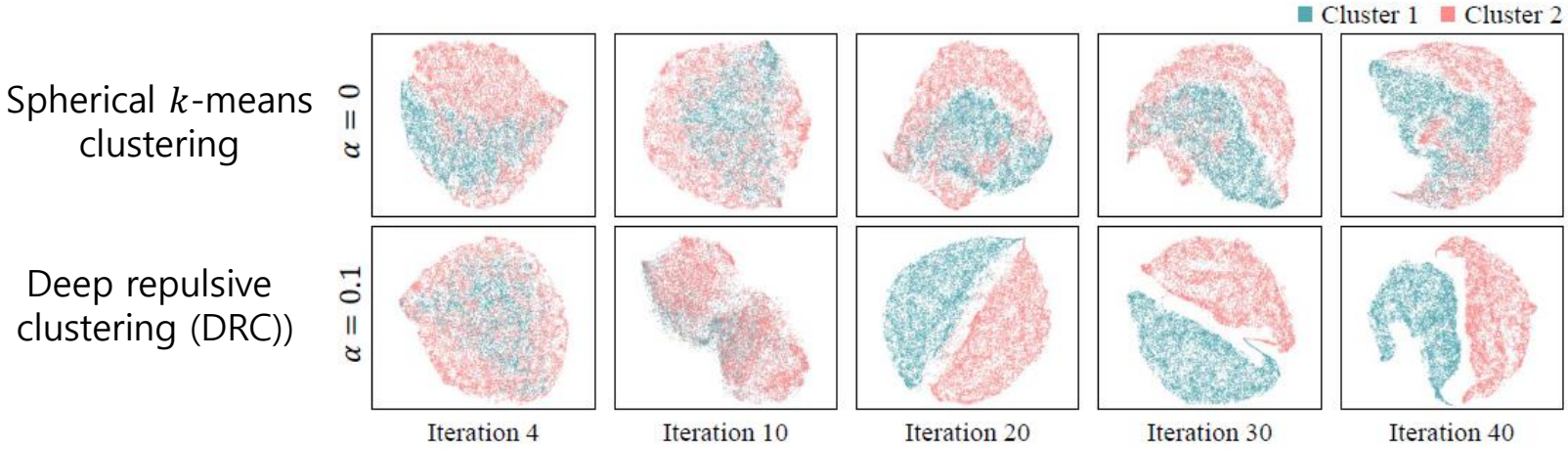
$$C_j = \left\{x \middle| (h_{id}^{x})^T c_j \geq (h_{id}^{x})^T c_l\right\} \quad \text{for all } 1 \leq l \leq k$$

Repeat until converge

Not only **gather** data into clusters,
Also repel each cluster to strictly **separate** them

3. Experimental results

- Facial age estimation
  - MORPH II dataset, d_{id}=896



[Comparison of the feature space transition of MORPH II (t-SNE visualization)]

[Clustering results after DRC (k=2)]

3. Experimental results

- Facial age estimation
  - MORPH II dataset, d_{id}=896

(Mean absolute error)

	Algorithm	MAE	CS (%)
Conventional age estimation algorithm	DRFs (Shen et al., 2018)	2.91	82.9
	MO-CNN* (Tan et al., 2017)	2.52	85.0
	MV (Pan et al., 2018)	-	-
	MV* (Pan et al., 2018)	-	-
	BridgeNet* (Li et al., 2019)	2.38	91.0
	AVDL* (Wen et al., 2020)	2.37	-
	OL* (Lim et al., 2020)	2.41	91.7
Proposed age estimation algorithm	Proposed-Vanilla ($k = 2$)	3.36	80.1
	Proposed-VGG ($k = 1$)*	<u>2.35</u>	<u>92.4</u>
	Proposed-VGG ($k = 2$)*	**2.26**	**93.8**

Algorithm	MAE	CS (%)
Proposed without repulsive term	2.47	90.7
Proposed with repulsive term	2.26	93.8

Proposed algorithm increases the accuracy of the age estimation!

Thank you

Catch & Carry: Reusable Neural Controllers for Vision-Guided Whole-Body Tasks

Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, Nicolas Heess, DeepMind

SIGGRAPH 2020

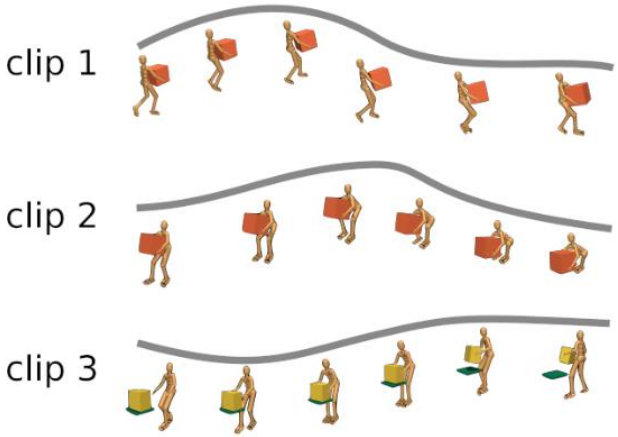
Presenter: Dohyeong Kim

Department of Electrical and Computer Engineering
Seoul National University

RLLAB
http://rllab.snu.ac.kr

Index

Overview



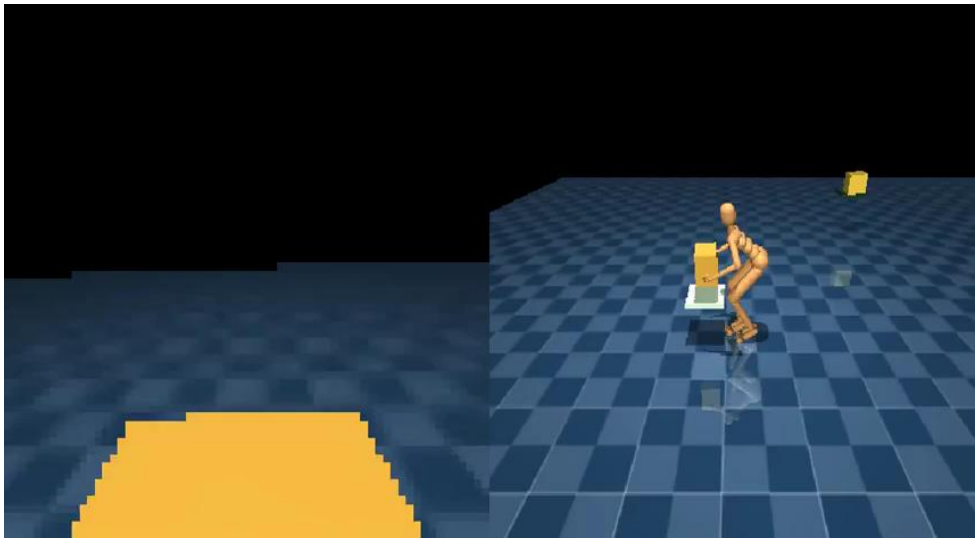
Motion capture data

Inverse dynamics + supervised learning

Low-level controller

Egocentric vision + Model-free RL

high-level controller



Inverse Dynamics from Motion Capture

Given dataset:

$$(\ldots, s_t, s_{t+1}, s_{t+2}, \ldots)$$

After inverse dynamics:

$$(\ldots, \hat{s}_t, a_t, \hat{s_{t+1}}, a_{t+1}, \hat{s_{t+2}}, \ldots)$$

Need to extract actions from state-only trajectories (inverse dynamics), but how?

➡ Train RL agents to track given motion capture data individually.

Reward for reference motion tracking:

$$r_t = \exp(-\beta E_{\text{total}} / w_{\text{total}}), \quad E_{\text{total}} = w_{\text{qpos}} E_{\text{qpos}} + w_{\text{qvel}} E_{\text{qvel}} + w_{\text{ori}} E_{\text{ori}} + w_{\text{app}} E_{\text{app}} + w_{\text{vel}} E_{\text{vel}} + w_{\text{gyro}} E_{\text{gyro}} + w_{\text{obj}} E_{\text{obj}},$$

$$E_{\text{ori}} = ||\log(\vec{q}_{\text{ori}} \cdot \vec{q}_{\text{ori}}^{\star -1})||_2 \quad E_{\text{qpos}} = \frac{1}{N_{\text{qpos}}} \sum |\vec{q}_{pos} - \vec{q}_{pos}^{\star}| \quad E_{\text{app}} = \frac{1}{N_{\text{app}}} \sum ||\vec{x}_{\text{app}} - \vec{x}_{\text{app}}^{\star}||_2$$

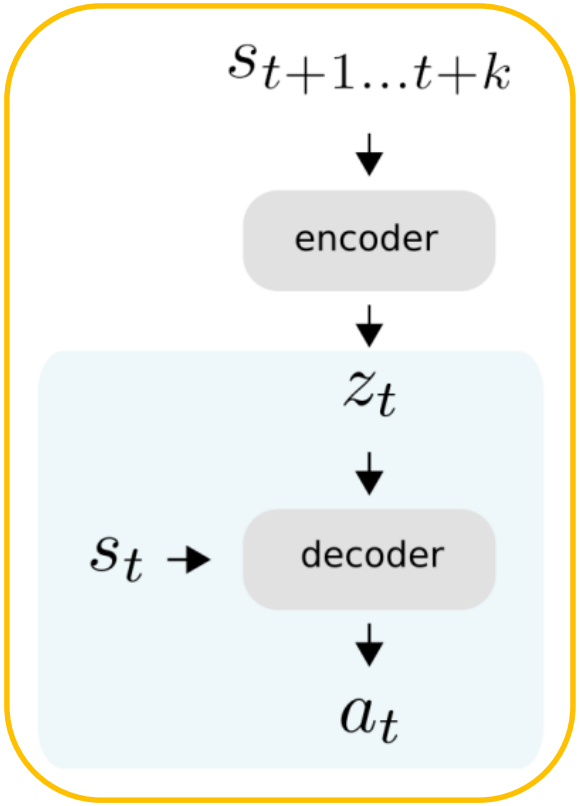
$$E_{\text{gyro}} = 0.1 \cdot ||\vec{q}_{\text{gyro}} - \vec{q}_{\text{gyro}}^{\star}||_2 \quad E_{\text{qvel}} = \frac{1}{N_{\text{qvel}}} \sum |\vec{q}_{\text{vel}} - \vec{q}_{\text{vel}}^{\star}| \quad E_{\text{vel}} = 0.1 \cdot \frac{1}{N_{\text{vel}}} \sum |\vec{x}_{\text{vel}} - \vec{x}_{\text{vel}}^{\star}|$$

$$E_{\text{obj}} = ||\vec{x}_{\text{obj}} - \vec{x}_{\text{obj}}^{\star}||_2$$

Low-Level Motor Skill Module

Low-level controller

Structure:



- Why use several future states?
  - To reduce ambiguity in motion.
  - Reflect short-term motor intention.
  - (k=5 in the paper.)
- Latent space is called skill embedding space.

Loss: $$\mathbb{E}_q\Big[\sum_{t=1}^{T} \log \pi(a_t|s_t,z_t) + \beta\big(\log p_z(z_t|z_{t-1}) - \log q(z_t|z_{t-1},s_{t+1\ldots t+k})\big)\Big]$$

decoder prior encoder

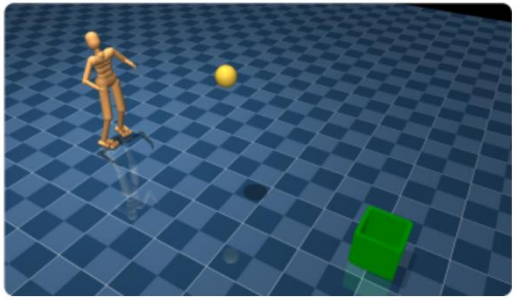
(Train with the action extracted trajectories.)

High-Level Task Policy

high-level controller

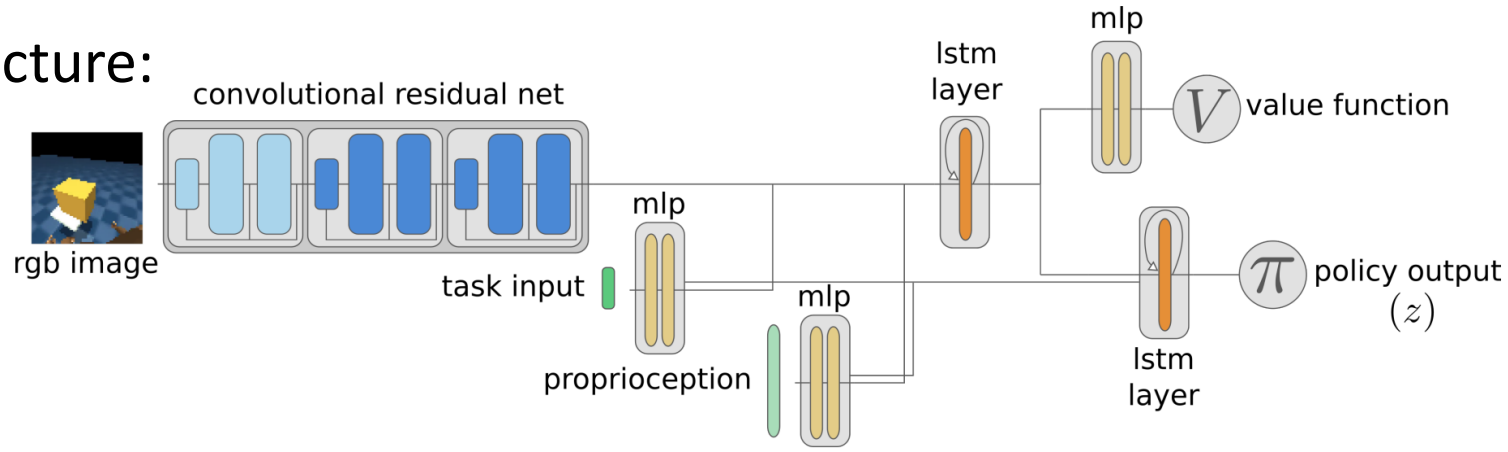
Tasks:

Warehouse



Toss

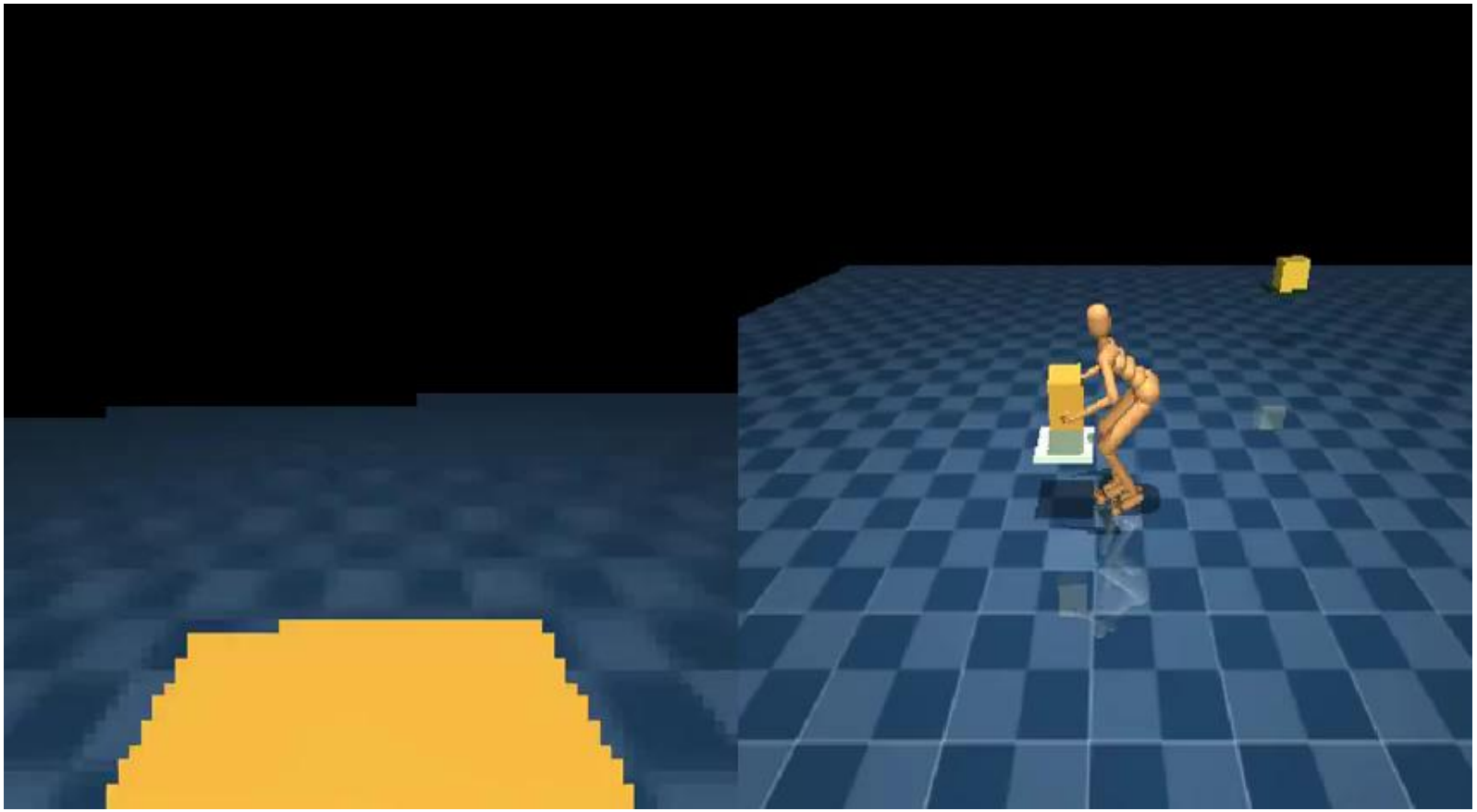
Structure:



Training: Model-free RL (Maximum a posteriori policy optimization)

+ sparse reward (+1 when task success, otherwise 0)

Result



Conclusion

1. Train RL agents interacting with objects from observations.
2. Use egocentric vision data for object perception.
3. Take so long training time & show artifacts.

RLLAB

http://rllab.snu.ac.kr