Large Loss Matters in Weakly Supervised Multi-Label Classification

Youngwook Kim [1*], Jae Myung Kim [2*], Zeynep Akata [2,3], Jungwoo Lee [1,4]

[1]Seoul National University, [2]University of Tübingen, [3]Max Plank Institute, [4]HodooAI Lab

Introduction

Annotating as a **partial label** can reduce annotation cost for multi-label classification.

It enables us to collect large-scale multi-label dataset with relatively small effort. (e.g. JFT-300M, InstagramNet-1B)

Assuming unobserved labels as negative (AN) introduces **noisy** supervision which may hamper the model learning.

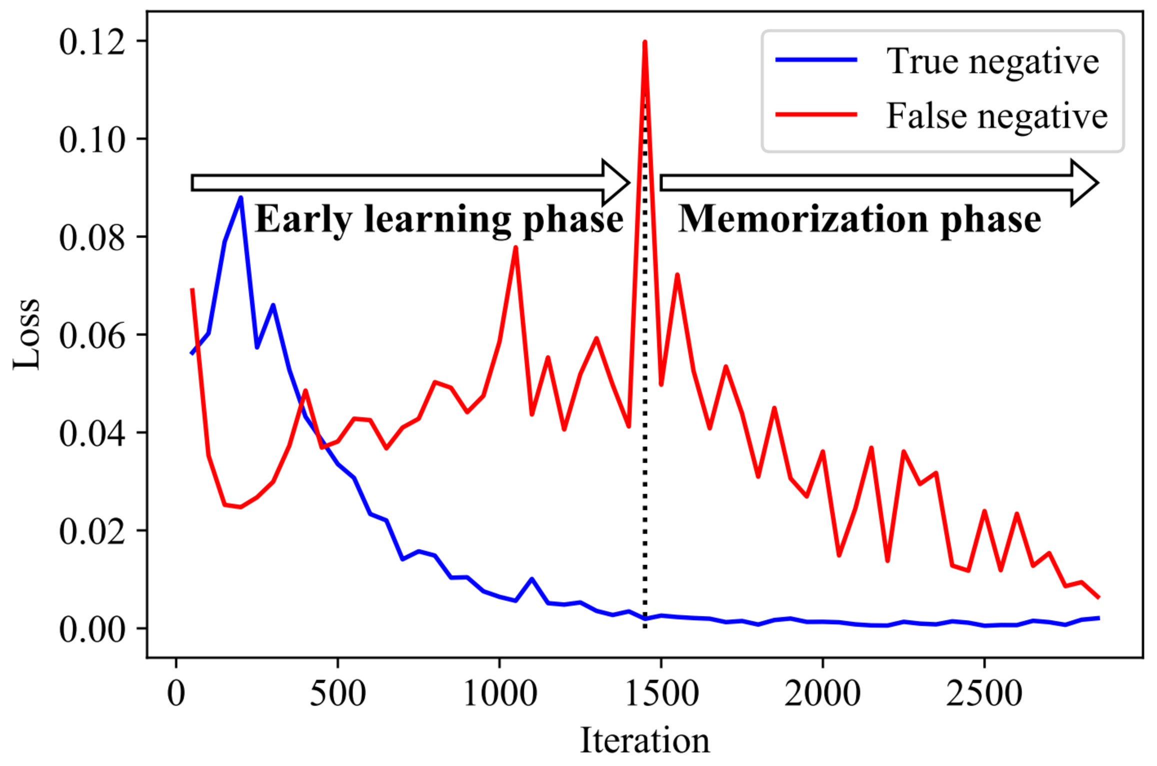
	[a]	[b]	[c]
car	✓	✓	✓
person	✓		✗
boat	✗		✗
bear	✗	✗	✗
apple	✗		✗

[a] : full label; [b] : partial label; [c] : AN target with **false negative**

Our key observation : Memorization effect

When training a model with noisy AN target, **the model first fits into clean label and then gradually fits into noisy label!**

We can discriminate whether a specific sample is noisy with its **loss value!** (before training finish)



Highest loss phase	Pascal VOC (%) TP	TN	FN	MS COCO (%) TP	TN	FN
Warmup	**88.3**	**90.7**	23.8	**64.0**	**82.6**	17.3
Regular	11.7	9.3	**72.2**	36.0	17.4	**82.7**

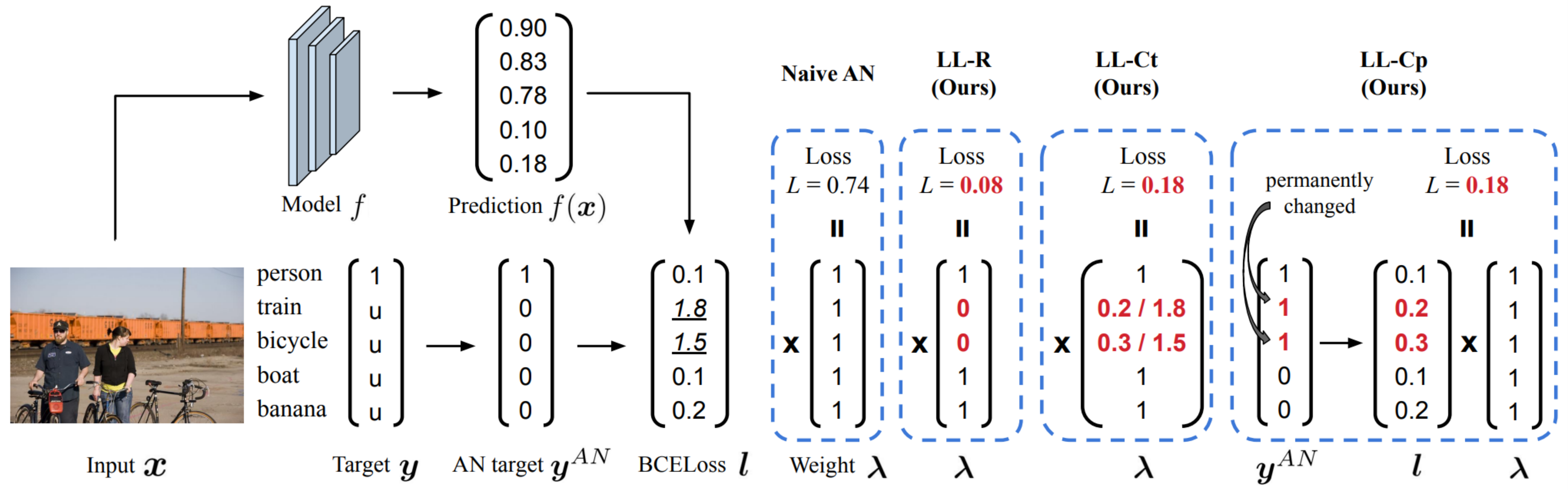
Methodology

Main idea : Reject or correct large loss samples during training!

Define AN target $y_i^{AN} = \begin{cases} 1, & i \in \mathcal{S}^p \\ 0, & i \in \mathcal{S}^n \cup \mathcal{S}^u \end{cases}$ where $\mathcal{S}^p = \{i \mid y_i = 1\}$, $\mathcal{S}^n = \{i \mid y_i = 0\}$, $\mathcal{S}^u = \{i \mid y_i = u\}$

Introduce the weight term λ_i in a standard BCE loss function

$$L = \frac{1}{|\mathcal{D}'|} \sum_{(\boldsymbol{x}, \boldsymbol{y}^{AN}) \in \mathcal{D}'} \frac{1}{K} \sum_{i=1}^{K} \mathrm{BCELoss}\left(f(\boldsymbol{x})_i, y_i^{AN}\right) \times \lambda_i$$



0) Naive AN (Vanilla BCE) : $\lambda_i = 1$ for all i

1) LL-R (Large Loss - Rejection) : $\lambda_i = \begin{cases} 0, & i \in \mathcal{S}^u \text{ and } l_i > R(t) \\ 1, & \text{otherwise} \end{cases}$

2) LL-Ct (Correction) (temporary) : $\lambda_i = \begin{cases} \frac{\log f(\boldsymbol{x})_i}{\log(1 - f(\boldsymbol{x})_i)}, & i \in \mathcal{S}^u \text{ and } l_i > R(t) \\ 1, & \text{otherwise} \end{cases}$

$R(t)$: Top $[(t-1) \cdot \Delta_{rel}]\%$ loss value in mini-batch at epoch t

3) LL-Cp (Correction) (permanent) : $\lambda_i = 1$ for all i

with $y_i^{AN} = \begin{cases} 1, & i \in \mathcal{S}^u \text{ and } l_i > R(t) \\ unchanged, & \text{otherwise} \end{cases}$

$R(t)$: Top $[\Delta_{rel}]\%$ loss value in mini-batch at epoch t

Results

Our proposed methods achieve **state-of-the-art** performance both on **artificially created** & **real** partial label datasets!

Method	End-to-end VOC	COCO	NUSWIDE	CUB	LinearInit. VOC	COCO	NUSWIDE	CUB
Full label	90.2	78.0	54.5	32.9	91.1	77.2	54.9	34.0
Naive AN	85.1	64.1	42.0	19.1	86.9	68.7	47.6	20.9
WAN [7, 28]	86.5	64.8	46.3	20.3	87.1	68.0	47.5	21.1
LSAN [7, 39]	86.7	66.9	44.9	17.9	86.5	69.2	50.5	16.6
EPR [7]	85.5	63.3	46.0	20.0	84.9	66.8	48.1	21.2
ROLE [7]	87.9	66.3	43.1	15.0	88.2	69.0	**51.0**	16.8
LL-R (Ours)	**89.2**	**71.0**	47.4	19.5	**89.4**	**71.9**	49.1	21.5
LL-Ct (Ours)	89.0	70.5	48.0	**20.4**	89.3	71.6	49.6	**21.8**
LL-Cp (Ours)	88.4	70.7	**48.3**	20.1	88.3	71.0	49.4	21.4

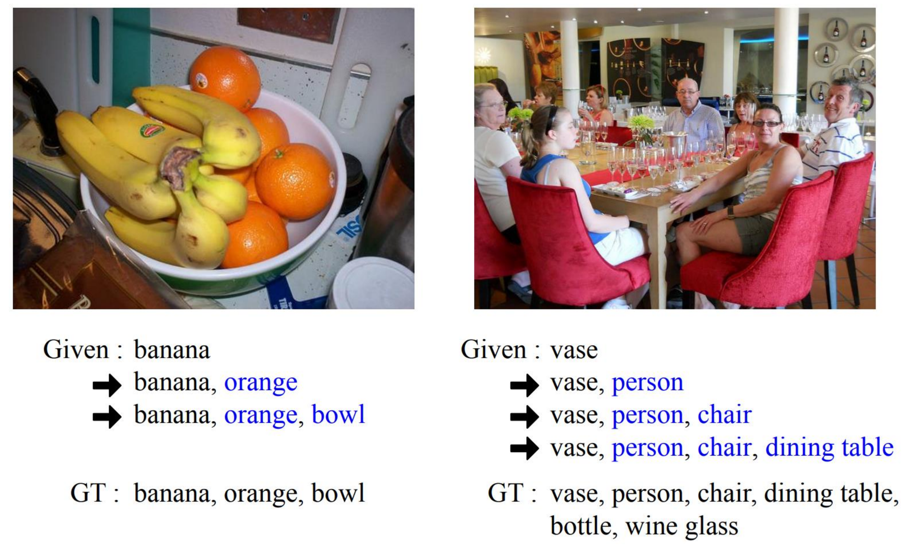
(OpenImages V3)

Method	G1	G2	G3	G4	G5	All Gs
Naive IU	69.5	70.3	74.8	79.2	85.5	75.9
Curriculum [9]	70.4	71.3	76.2	80.5	86.8	77.1
IMCL [16]	71.0	72.6	77.6	81.8	87.3	78.1
Naive AN	77.1	78.7	81.5	84.1	88.8	82.0
WAN [7, 28]	71.8	72.8	76.3	79.7	84.7	77.0
LSAN [7, 39]	68.4	69.3	73.7	77.9	85.6	75.0
LL-R (Ours)	77.4	79.1	82.0	84.5	89.5	82.5
LL-Ct (Ours)	77.7	79.3	82.1	84.7	89.4	**82.6**
LL-Cp (Ours)	77.6	79.1	81.9	84.6	89.4	82.5

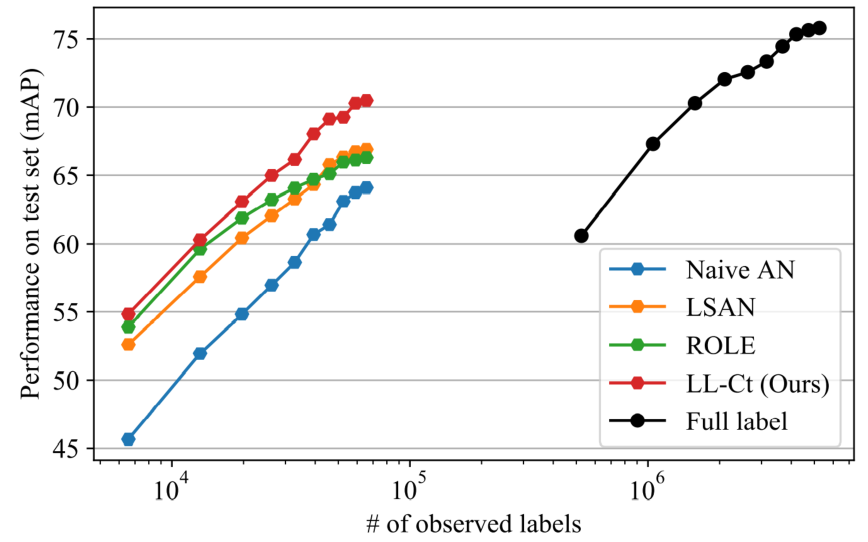
More analyses…

Qualitative results

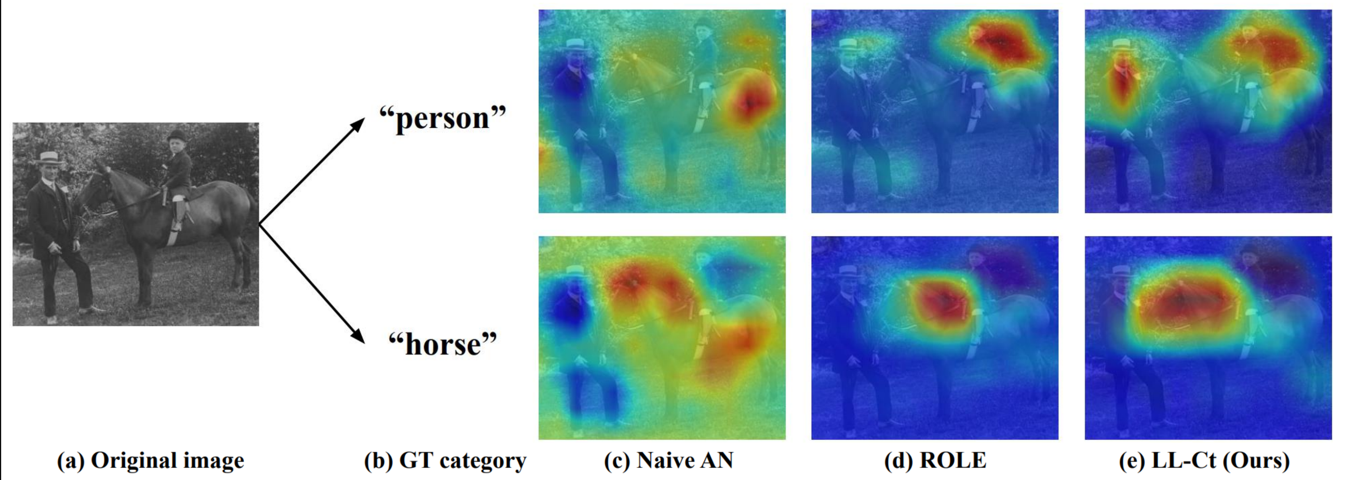
(corrected labels on LL-Ct)



Labelling efficiency



Model explanation



Pointing game

Method	VOC	COCO
Naive AN	78.9	46.4
WAN [7, 28]	79.8	47.7
LSAN [7, 39]	79.5	49.1
EPR [7]	80.2	48.1
ROLE [7]	82.5	51.5
LL-R (Ours)	**83.7**	54.0
LL-Ct (Ours)	**83.7**	**54.1**
LL-Cp (Ours)	83.5	53.3

Our code is also available at