Multi-class classification without Multi-class Labels

Yen-Chang Hsu[1], Zhaoyang Lv[1], Joel Schlosser[2], Phillip Odom[2], and Zsolt Kira[12]
[1]Georgia Institute of Technology
[2]Georgia Tech Research Institute
[1]{yenchang.hsu,zhaoyang.lv,zkira}@gatech.edu
[2]{joel.schlosser,phillip.odom}@gtri.gatech.edu

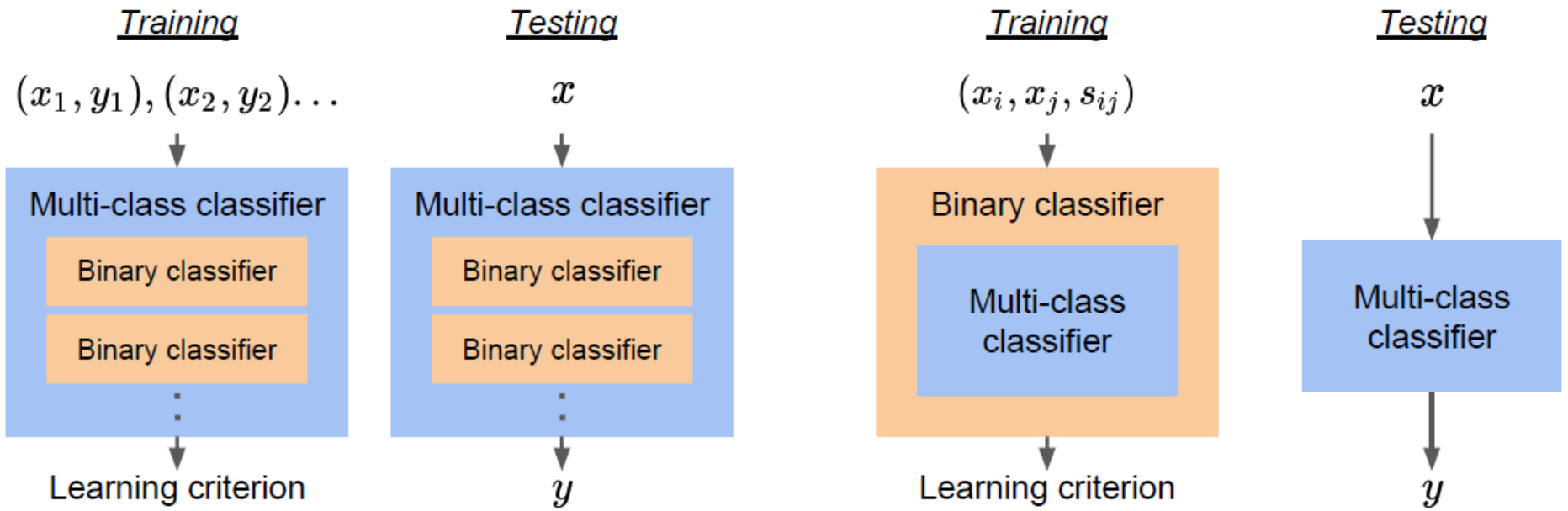
ICLR 2019

Motivation

- A neural network demands a large amount of class-specific labels for learning a discriminative model, this type of labeling can be expensive to collect.

- Pairwise similarity between examples, which is a weaker form of annotation, is easily to collect.

Meta classification learning



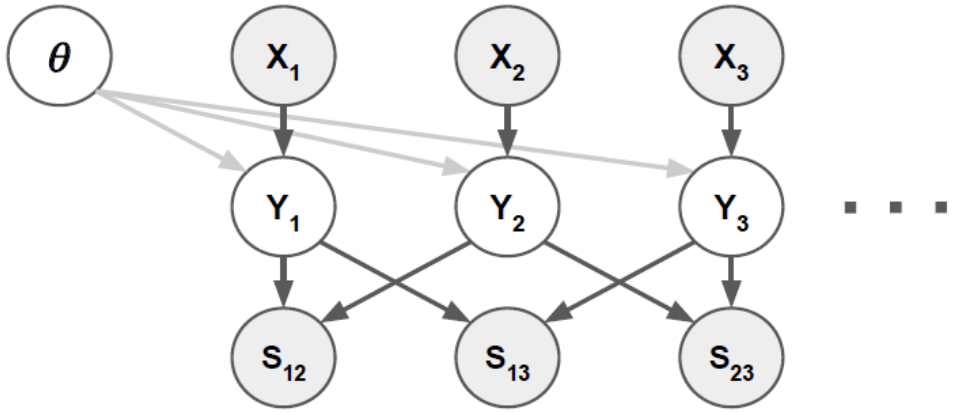
(a) Binary classifier ensemble

(b) Meta classification learning

Figure 1: Problem reduction schemes for multi-class classification. This work proposes scheme (b), which introduces a binary classifier that captures s_{ij}. Note that s_{ij} represents the probability that x_i and x_j belong to the same class.

Meta classification learning

- Graphical representation for meta classification task



Additional independences:

$$S_{ij} \perp \mathbf{S} \backslash \{S_{ij}\} | X_i, X_j$$

- likelihood

$$P(S|Y) = \prod_{i,j} p(S_{ij}|Y_i, Y_j)$$

$$L(\theta; X, Y, S) = P(X, Y, S; \theta) = P(S|Y)P(Y|X; \theta)P(X)$$

Meta classification learning

- Compute the likelihood

$$\mathcal{L}(\theta; \mathbf{X}, \mathbf{S}) \approx \sum_{\mathbf{Y}} \mathrm{P}(\mathbf{S}|\mathbf{Y})\mathrm{P}(\mathbf{Y}|\mathbf{X};\theta)$$

$$\approx \prod_{i,j} \Big(\sum_{Y_i=Y_j} \mathbb{1}[s_{ij}=1]\mathrm{P}(Y_i|x_i;\theta)\mathrm{P}(Y_j|x_j;\theta) + \sum_{Y_i\neq Y_j} \mathbb{1}[s_{ij}=0]\mathrm{P}(Y_i|x_i;\theta)\mathrm{P}(Y_j|x_j;\theta) \Big).$$

- Loss Function

$$L_{meta}(\theta) = -\sum_{i,j} \log \Big(\sum_{Y_i=Y_j} \mathbb{1}[s_{ij}=1]\mathrm{P}(Y_i|x_i;\theta)\mathrm{P}(Y_j|x_j;\theta) + \sum_{Y_i\neq Y_j} \mathbb{1}[s_{ij}=0]\mathrm{P}(Y_i|x_i;\theta)\mathrm{P}(Y_j|x_j;\theta) \Big)$$

$$= -\sum_{i,j} s_{ij} \log(f(x_i;\theta)^T f(x_j;\theta)) + (1-s_{ij}) \log(1 - f(x_i;\theta)^T f(x_j;\theta)).$$

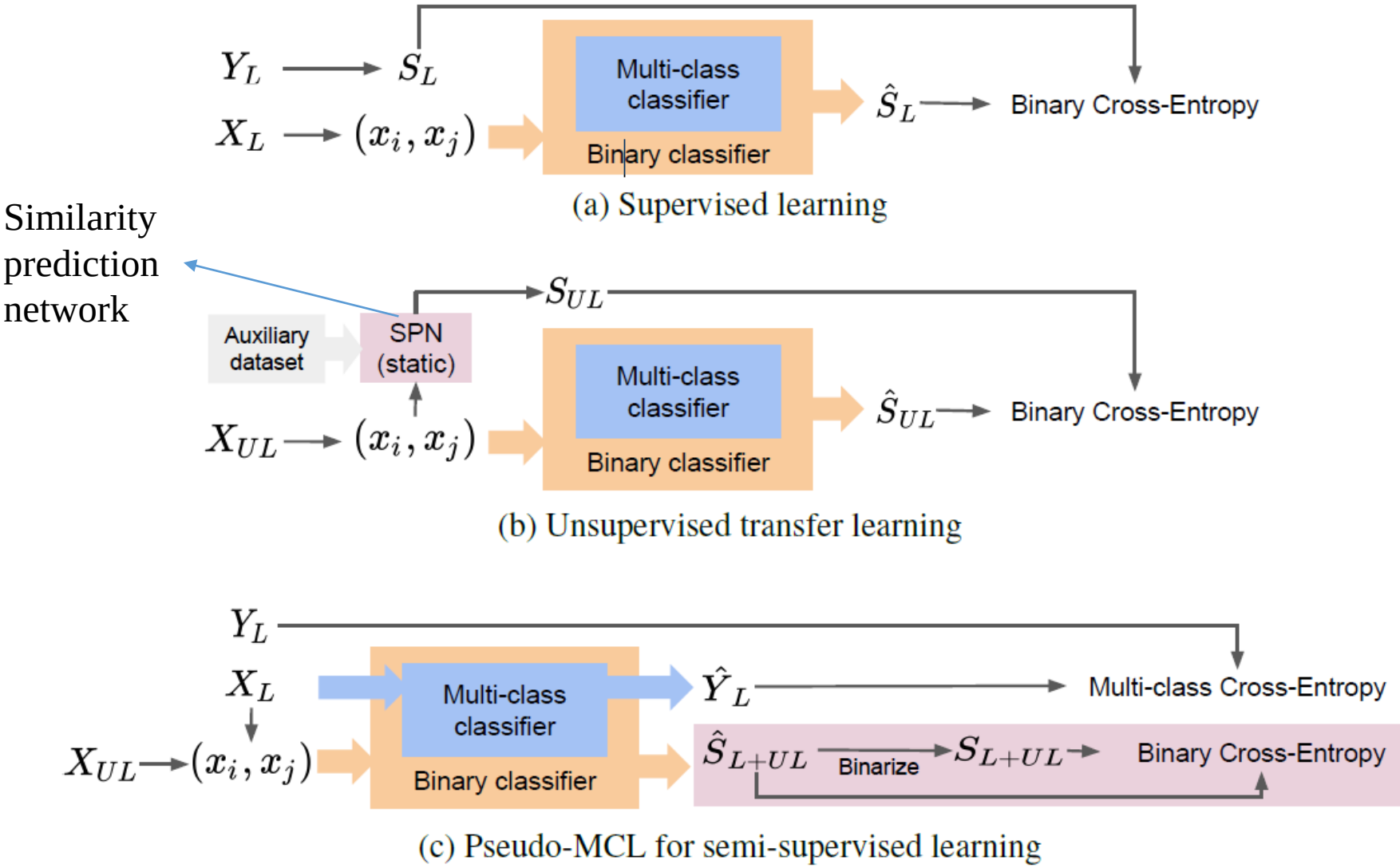
$$L_{meta} = -\sum_{i,j} s_{ij} \log \hat{s}_{ij} + (1-s_{ij}) \log(1-\hat{s}_{ij}).$$

$$g(x_i, x_j, f(\cdot,\theta)) = f(x_i;\theta)^T f(x_j;\theta) = \hat{s}_{ij}$$

Function g :the probability of having the same class label:

$\hat{S}_{i,j}$: *the predicted similarity*

Learning paradigms



Experiment

KCL:KLD-based Contrastive Loss(KCL)

- Cost between a similar pair:

$$L^{+}_{KCL}(x_i, x_j) = D_{\mathrm{KL}}(\hat{\mathbf{y}}_i || \hat{\mathbf{y}}_j) + D_{\mathrm{KL}}(\hat{\mathbf{y}}_j || \hat{\mathbf{y}}_i).$$

- Cost between a dissimilar pair:

$$L^{-}_{KCL}(x_i, x_j) = L_h(D_{\mathrm{KL}}(\hat{\mathbf{y}}_i || \hat{\mathbf{y}}_j), \sigma) + L_h(D_{\mathrm{KL}}(\hat{\mathbf{y}}_j || \hat{\mathbf{y}}_i), \sigma),$$
$$\text{where } L_h(e, \sigma) = max(0, \sigma - e).$$

- Total contrastive loss(KCL):

$$L_{KCL} = \sum_{i,j} s_{ij} L^{+}_{KCL}(x_i, x_j) + (1 - s_{ij}) L^{-}_{KCL}(x_i, x_j).$$

Supervised learning with weak labels

Table 1: The classification error rate (lower is better) on three datasets with different objective functions and different neural network architectures. CE denotes that the network uses class-specific labels for training with a multi-class cross-entropy. MCL only uses the binarized similarity for learning with the meta-classification criterion. KCL is a strong baseline which also uses binarized similarity. The * symbol indicates the worst cases of KCL. The performance in parenthesis means its network uses a better initialization (VGG16 and VGG8) or a learning schedule which is 10 times longer (VGG11). The two treatments are discussed in Section 5.2.1. We only use VGG8 for CIFAR100 since KCL performs the best with it on CIFAR10. Each value is the average of 3 runs.

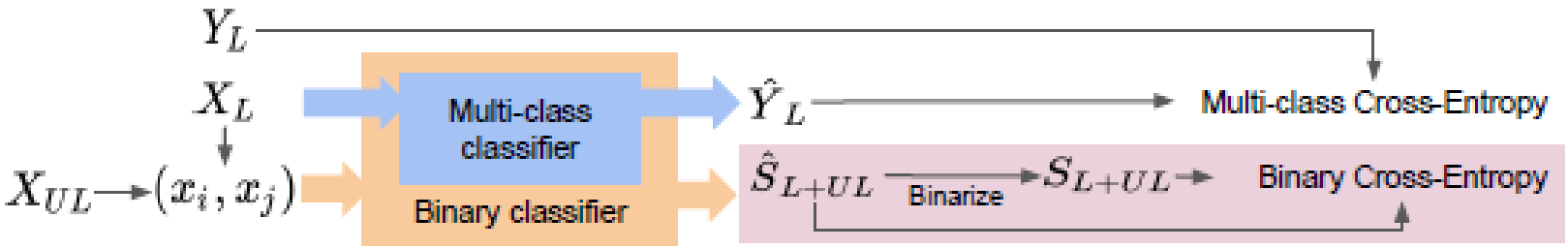
Dataset	#class	Network	(Class label) CE	(Pairwise label) KCL	(Pairwise label) MCL
MNIST	10	LeNet	0.6%	**0.5**%	0.6%
CIFAR10	10	LeNet	14.9%	16.4%	15.1%
		VGG8	10.2%	10.2%	10.2%
		VGG11	8.9%	72.2(10.4)%	9.4%
		VGG16	7.6%	*81.1(10.3)%	8.3%
		ResNet18	6.7%	73.8%	6.6%
		ResNet34	6.6%	79.3%	6.3%
		ResNet50	6.6%	79.6%	5.9%
		ResNet101	6.5%	79.9%	**5.6**%
CIFAR100	100	VGG8	**35.4**%	*45.3(40.2)%	36.1%

Unsupervised cross-task transfer learning

Table 2: Unsupervised cross-task transfer learning on Omniglot. The performance (higher is better) is averaged across 20 alphabets (datasets), in which each has 20 to 47 letters (classes). The ACC and NMI without brackets have the number of output nodes K equal to the true number of classes in a dataset, while columns with "(K=100)" represent the case where the number of classes is unknown and a fixed $K = 100$ is used.

Method	ACC	ACC (K=100)	NMI	NMI (K=100)
K-means (MacQueen et al., 1967)	21.7%	18.9%	0.353	0.464
LPNMF (Cai et al., 2009)	22.2%	16.3%	0.372	0.498
LSC (Chen & Cai, 2011)	23.6%	18.0%	0.376	0.500
ITML (Davis et al., 2007)	56.7%	47.2%	0.674	0.727
SKKm (Anand et al., 2014)	62.4%	46.9%	0.770	0.781
SKLR (Amid et al., 2016)	66.9%	46.8%	0.791	0.760
CSP (Wang et al., 2014)	62.5%	65.4%	0.812	0.812
MPCK-means (Bilenko et al., 2004)	81.9%	53.9%	0.871	0.816
KCL (Hsu et al., 2018)	82.4%	78.1%	0.889	0.874
MCL (ours)	**83.3%**	**80.2%**	**0.897**	**0.893**

Semi-supervised learning



(c) Pseudo-MCL for semi-supervised learning

Table 4: Test error rates (lower is better) obtained by various semi-supervised learning approaches on CIFAR-10 with all but 4,000 labels removed. Supervised refers to using only 4,000 labeled samples from CIFAR-10 without any unlabeled data. All the methods use ResNet-18 and standard data augmentation.

Method	CIFAR10 4k labels
Supervised	$25.4 \pm 1.0\%$
Pseudo-Label	$19.8 \pm 0.7\%$
Π-model	$19.6 \pm 0.4\%$
VAT	$18.2 \pm 0.4\%$
SPN-MCL	$22.8 \pm 0.5\%$
Pseudo-MCL	$\mathbf{18.0} \pm 0.4\%$

Thanks