Deep Learning on Small Datasets without Pre-Training using Cosine Loss

-WACV 2020-

김병조

2020.04.22

Introduction

- Small Dataset
  - 학습 데이터 많이 모으기에는 한계 존재
  - Pre-trained model + fine-tuning으로 극복 가능
- Pre-trained model
  - ImageNet과 target domain의 차이 (target domain이 의학 이미지와 같이 특이한 경우)
  - License 문제
- **Small data without external information**
  - softmax + cross-entropy loss -> cosine loss
  - Small dataset: 20 ~ 100 images per class

Cosine Loss

- Cosine Similarity
  - $\sigma_{cos}(a,b) = \cos(a\angle b) = \frac{<a,b>}{\|a\|_2 \|b\|_2}$
- Cosine loss function
  - $f_\theta : X \to R^d, \psi : R^d \to P, \varphi : C \to P$
  - $L_{cos}(x,y) = 1 - \sigma_{cos}(f_\theta(x), \varphi(y))$
- Cosine loss function with unit hypersphere
  - $\psi = \frac{x}{\|x\|_2}, \varphi_{onehot}(y) = [0 \ldots 1 \ldots 0]$
  - $L_{cos}(x,y) = 1 - < \varphi_{onehot}(y), \psi\big(f_\theta(x)\big) >$

vs Categorical Cross-Entropy & Mean Squared Error

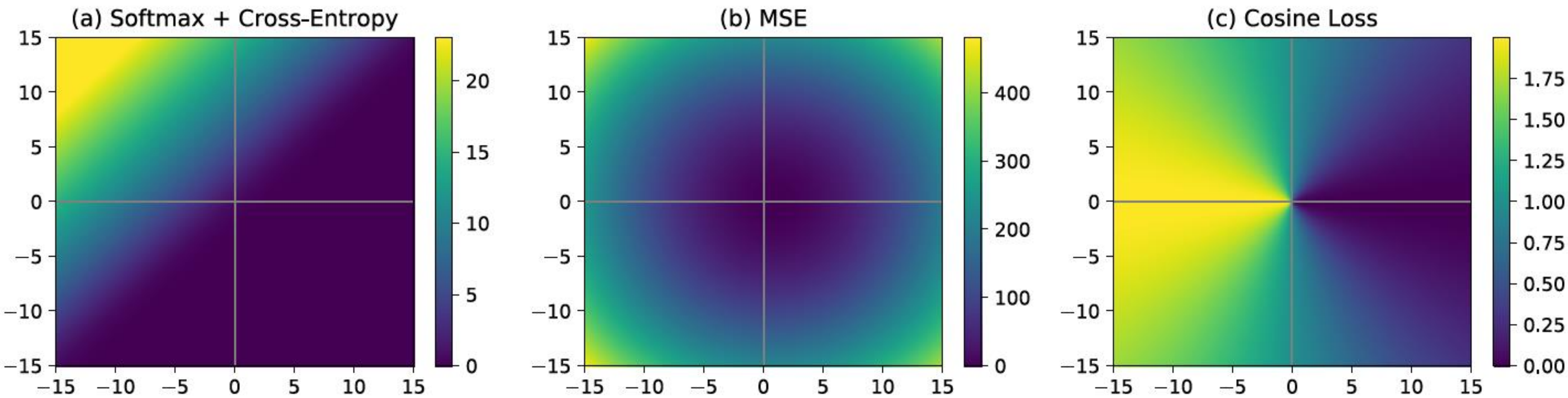

Figure 1: Heatmaps of three loss functions in a 2-D feature space with fixed target $\varphi(y) = \begin{bmatrix} 1 & 0 \end{bmatrix}^\top$.

- Cosine loss의 경우 [0, 2] 안에 loss 값이 존재
- Direction 만을 고려 -> scaling에 invariant

vs Categorical Cross-Entropy & Mean Squared Error

- Cross Entropy loss
  - 급강하 영역
  - 넓은 영역
  - 각 영역 안은 매우 차이가 적음
  - 초기화 및 learning rate 설정 중요

  -> Cosine loss는 색이 고르게 분포되어 있어서 더 robust 할 것!

vs Categorical Cross-Entropy & Mean Squared Error

- Cross Entropy loss
  - True class 값이 다른 class 보다 매우 커야만 loss가 작다. [0.001, 0.0001, **0.991**, 0.0001 …]
  - small data일 때 overfiting 일어난다.
  - label smoothing 적용하여 해결한다. (hyper-parameter 사용)

  -> Cosine loss는 unit hypersphere 만들 때 L2 normalization으로 regularization (hyper-parameter 없이)

  -> 또한 클래스 하나에 국한되는게 아님. [0.2, 0.58, 0.21 …]

- Mean Squared Error
  - Euclidean distance 사용
  - 높은 차원일 때 문제 (curse of dimension)

  -> Cosine loss는 direction만을 고려

Semantic Class Embeddings

- one-hot vector에는 semantic relationship이 고려 안됨.
- Wordnet과 같은 ontology 이용하여 class embedding φ_{sem} (https://arxiv.org/abs/1809.09924)
- Semantic relationship이 추가되면서 분류 정확성 위해 cross entropy loss 추가
- g_θ: softmax + fully-connected layer
- $$\boldsymbol{L_{cos+xent}(x,y) = 1 - < \varphi_{sem}(y), \psi\big(f_\theta(x)\big) > - \lambda < \varphi_{onehot}(y), \log(g_\theta\Big(\psi\big(f_\theta(x)\big)\Big)) >}$$

Semantic Class Embeddings

- one-hot vector에는 semantic relationship이 고려 안됨.
- Wordnet과 같은 ontology 이용하여 class embedding φ_{sem} (https://arxiv.org/abs/1809.09924)
- Semantic relationship이 추가되면서 분류 정확성 위해 cross entropy loss 추가
- g_θ: softmax + fully-connected layer
- $$\boldsymbol{L_{cos+xent}(x,y) = 1- < \varphi_{sem}(y), \psi\big(f_\theta(x)\big) > - \lambda \boxed{< \varphi_{onehot}(y), \log(g_\theta\Big(\psi\big(f_\theta(x)\big)\Big)) >}}$$

Cross-Entropy loss와 동일

Ex.) <(0, 1, 0), (log(0.1), log(0.8), log(0.1)> = 0*log(0.1) + 1*log(0.8) + 0*log(0.1)

Experiments

Dataset	#Classes	#Training	#Test	Samples/Class
CUB	200	5,994	5,794	29 – 30 (30)
NAB	555	23,929	24,633	4 – 60 (44)
Cars	196	8,144	8,041	24 – 68 (42)
Flowers-102	102	2,040	6,149	20
MIT Indoor	67	5,360	1,340	77 – 83 (80)
CIFAR-100	100	50,000	10,000	500

Table 1: Image dataset statistics. The number of samples per class refers to training samples and numbers in parentheses specify the median.

Experiments

	CUB	NAB	Cars	Flowers-102	MIT Indoor	CIFAR-100
MSE	42.0	27.7	41.8	63.0	38.2	75.1
softmax + cross-entropy	51.9	59.4	78.2	67.3	44.3	77.0
softmax + cross-entropy + label smoothing	55.5	68.3	78.1	66.8	38.7	**77.5**
cosine loss (one-hot embeddings)	67.6	71.7	84.3	**71.1**	51.5	75.3
cosine loss + cross-entropy (one-hot embeddings)	**68.0**	**71.9**	**85.0**	70.6	**52.7**	76.4
cosine loss (semantic embeddings)	59.6	72.1	—	—	—	74.6
cosine loss + cross-entropy (semantic embeddings)	70.4	73.8	—	—	—	76.7
fine-tuned softmax + cross-entropy	82.5	80.1	91.2	97.2	79.9	—
fine-tuned cosine loss (one-hot embeddings)	82.7	78.6	89.6	96.2	74.3	—
fine-tuned cosine loss + cross-entropy (one-hot embeddings)	82.7	81.2	90.9	96.2	73.3	—

Table 2: Test-set classification accuracy in percent (%) achieved with different loss functions on various datasets. The best value per column not using external data or information is set in bold font.

Experiments

Embedding	Levels	$\mathcal{L}_{\text{cos}}$	$\mathcal{L}_{\text{cos+xent}}$
one-hot	1	**67.6**	68.0
flat	4	66.6	68.8
Wikispecies	4-6	61.6	69.9
deep	7	59.9	**70.4**

Table 3: Accuracy in % on the CUB test set obtained by cosine loss with class embeddings derived from taxonomies of varying depth. The best value per column is set in bold.

Semantic embedding 사용하면 Lcos 성능은 낮음 (분류 정확성이 낮음)
-> CE 사용하면 높아짐
깊은 계층구조를 가질수록 Lcos+xent 성능 높아짐
Semantic embedding은 유사한 클래스는 가까이, 안유사한 클래스는 멀리
-> dissimilar class의 고려를 더 할 수 있도록

Experiments

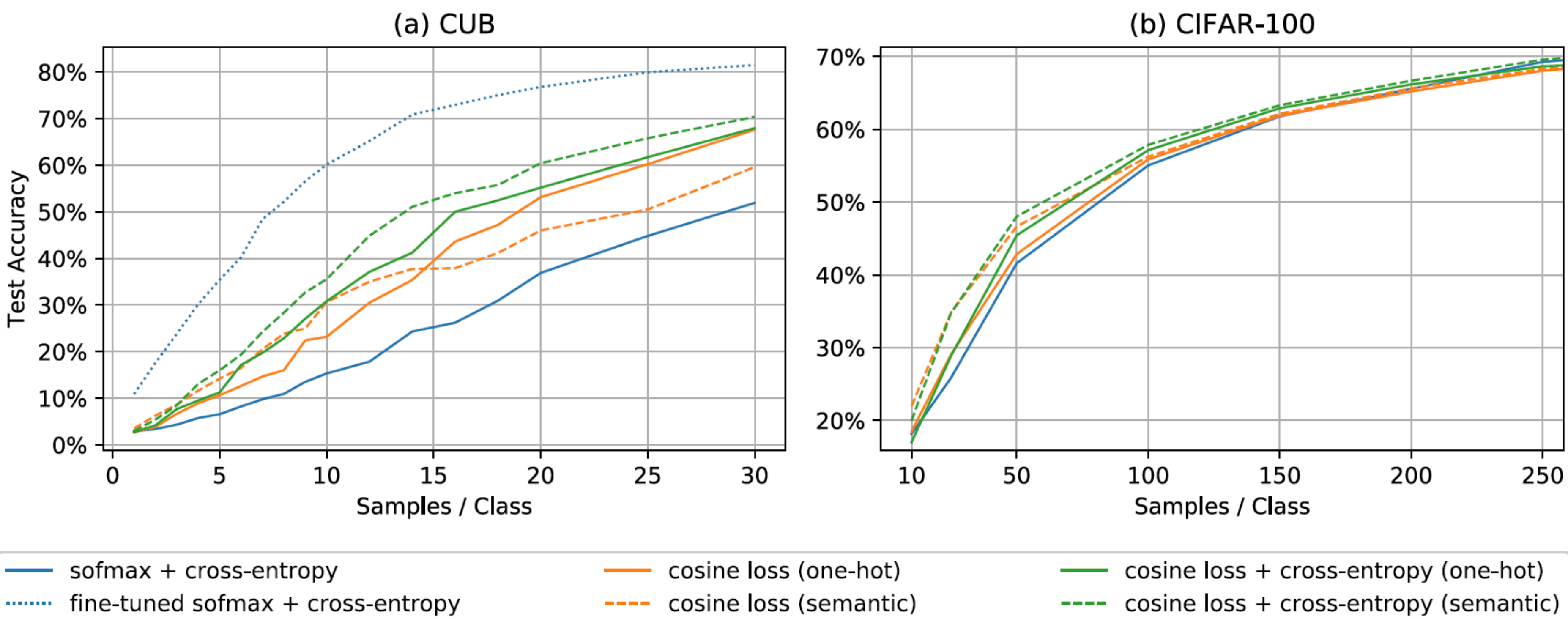

Figure 2: Classification performance depending on the dataset size.

Cosine loss는 CE loss 보다 더 나은 성능
Semantic embedding + CE 사용하면 더 가파른 성능 향상
당연히 fine-tuned model 사용하면 더 좋은 성능

Discussion (about VIPriors I.C.)

Model	Baseline (90)	Cosine+0.1CE (90/180)	Cosine+0.1CE+RandAugment (90/180)	Cutmix+CutmixCE (90,180, 270)
ResNet50	28.212	34.012/34.552	29.714/30.194	28.168/32.196/31.524
ResNet50 FConv	32.668	32.538/34.988		

더 추가해 볼 만한 기법..............?

Base model 변경...? (ex., EfficientNet)