Metric Learning for Landmark Retrieval Survey (Harry)

Query image

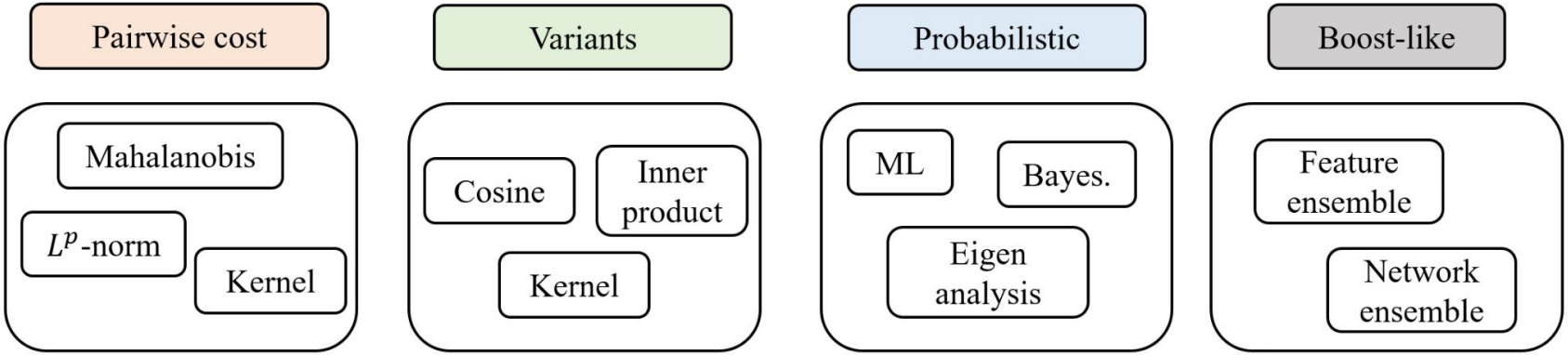
1. Query image

Database

2. Retrieve matched images

3. Top 4 matched images

⟨Pedigree of metric learning⟩



* Refer “Survey and experimental study on metric learning methods”, *Neural Networks*, 2018.

Overview Data Code Discussion Leaderboard Rules

Overview

- Description
- Evaluation
- Timeline
- Prizes
- Code Requirements
- ECCV 2020

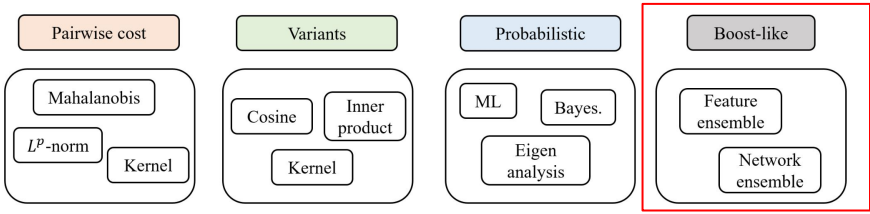
Welcome to the third Landmark Retrieval competition! This year, we have worked to set this up as a code competition and we have completely refreshed the test and index image sets.

Image retrieval is a fundamental problem in computer vision: given a query image, can you find similar images in a large database? This is especially important for query images containing landmarks, which accounts for a large portion of what people like to photograph.

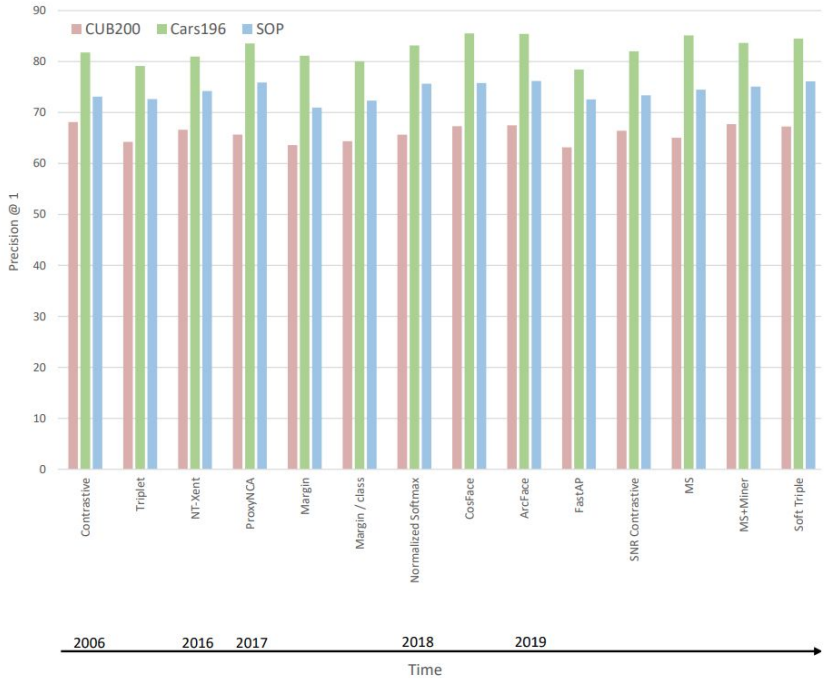
In this competition, the developed models are expected to retrieve relevant database images to a given query image (ie, the model should retrieve database images containing the same landmark as the query). This challenge is organized in conjunction with the Landmark Recognition Challenge 2020. Both challenges will be discussed at the Instance-Level Recognition workshop in ECCV'20.

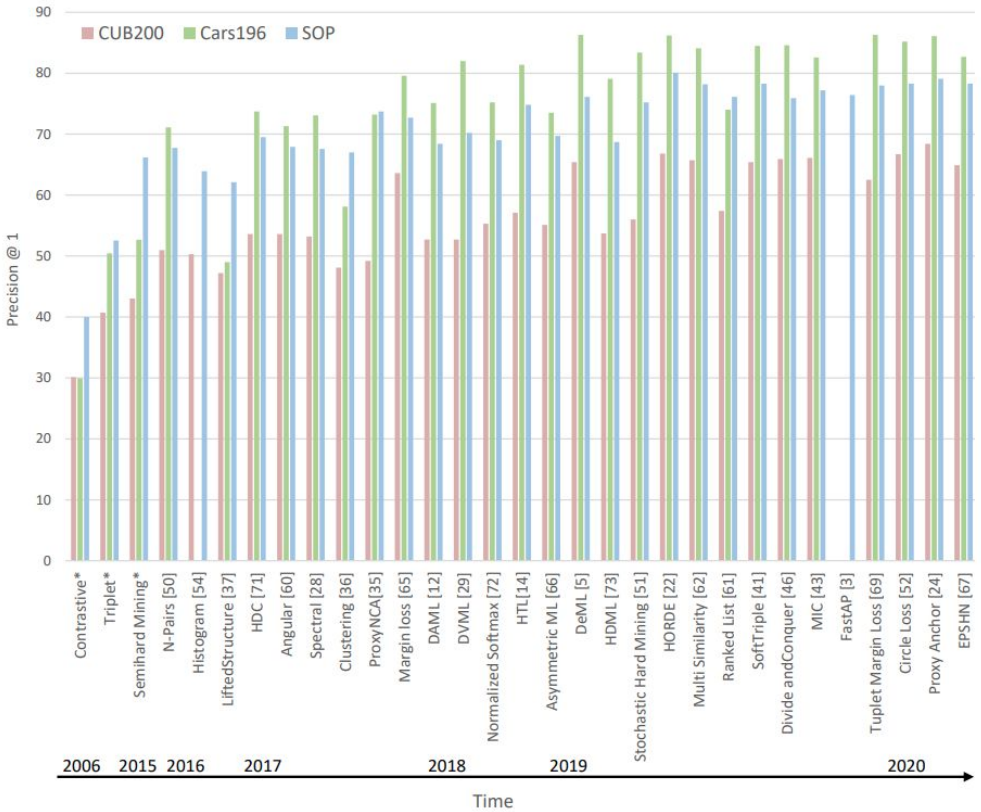
Computational burden!!

⟨Pedigree of metric learning⟩



* Refer "Survey and experimental study on metric learning methods", *Neural Networks*, 2018.


CUB200
Cars196
SOP
Precision @ 1
90
80
70
60
50
40
30
20
10
0
Contrastive
Triplet
NT-Xent
ProxyNCA
Margin
Margin / class
Normalized Softmax
CosFace
ArcFace
FastAP
SNR Contrastive
MS
MS+Miner
Soft Triple
2006
2016
2017
2018
2019
Time


CUB200
Cars196
SOP
Precision @ 1
90
80
70
60
50
40
30
20
10
0
Contrastive*
Triplet*
Semihard Mining*
N-Pairs [50]
Histogram [54]
LiftedStructure [37]
HDC [71]
Angular [60]
Spectral [28]
Clustering [36]
ProxyNCA[35]
Margin loss [65]
DAML [12]
DVML [29]
Normalized Softmax [72]
HTL[14]
Asymmetric ML [66]
DeML [5]
HDML [73]
Stochastic Hard Mining [51]
HORDE [22]
Multi Similarity [62]
Ranked List [61]
SoftTriple [41]
Divide andConquer [46]
MIC [43]
FastAP [3]
Tuplet Margin Loss [69]
Circle Loss [52]
Proxy Anchor [24]
EPSHN [67]
2006
2015
2016
2017
2018
2019
2020
Time

A Metric Learning Reality Check

Kevin Musgrave[1], Serge Belongie[1], Ser-Nam Lim[2]

[1]Cornell Tech [2]Facebook AI

Abstract. Deep metric learning papers from the past four years have consistently claimed great advances in accuracy, often more than doubling the performance of decade-old methods. In this paper, we take a closer look at the field to see if this is actually true. We find flaws in the experimental methodology of numerous metric learning papers, and show that the actual improvements over time have been marginal at best.

Table 4. Accuracy on CUB200

	Concatenated (512-dim)			Separated (128-dim)		
	P@1	**RP**	**MAP@R**	**P@1**	**RP**	**MAP@R**
Pretrained	51.05	24.85	14.21	50.54	25.12	14.53
Contrastive	**68.13 ± 0.31**	37.24 ± 0.28	26.53 ± 0.29	59.73 ± 0.40	31.98 ± 0.29	21.18 ± 0.28
Triplet	64.24 ± 0.26	34.55 ± 0.24	23.69 ± 0.23	55.76 ± 0.27	29.55 ± 0.16	18.75 ± 0.15
NT-Xent	66.61 ± 0.29	35.96 ± 0.21	25.09 ± 0.22	58.12 ± 0.23	30.81 ± 0.17	19.87 ± 0.16
ProxyNCA	65.69 ± 0.43	35.14 ± 0.26	24.21 ± 0.27	57.88 ± 0.30	30.16 ± 0.22	19.32 ± 0.21
Margin	63.60 ± 0.48	33.94 ± 0.27	23.09 ± 0.27	54.78 ± 0.30	28.86 ± 0.18	18.11 ± 0.17
Margin/class	64.37 ± 0.18	34.59 ± 0.16	23.71 ± 0.16	55.56 ± 0.16	29.32 ± 0.15	18.51 ± 0.13
N. Softmax	65.65 ± 0.30	35.99 ± 0.15	25.25 ± 0.13	58.75 ± 0.19	31.75 ± 0.12	20.96 ± 0.11
CosFace	67.32 ± 0.32	**37.49 ± 0.21**	**26.70 ± 0.23**	59.63 ± 0.36	31.99 ± 0.22	21.21 ± 0.22
ArcFace	67.50 ± 0.25	37.31 ± 0.21	26.45 ± 0.20	**60.17 ± 0.32**	**32.37 ± 0.17**	**21.49 ± 0.16**
FastAP	63.17 ± 0.34	34.20 ± 0.20	23.53 ± 0.20	55.58 ± 0.31	29.72 ± 0.16	19.09 ± 0.16
SNR	66.44 ± 0.56	36.56 ± 0.34	25.75 ± 0.36	58.06 ± 0.39	31.21 ± 0.28	20.43 ± 0.28
MS	65.04 ± 0.28	35.40 ± 0.12	24.70 ± 0.13	57.60 ± 0.24	30.84 ± 0.13	20.15 ± 0.14
MS+Miner	67.73 ± 0.18	37.37 ± 0.19	26.52 ± 0.18	59.41 ± 0.30	31.93 ± 0.15	21.01 ± 0.14
SoftTriple	67.27 ± 0.39	37.34 ± 0.19	26.51 ± 0.20	59.94 ± 0.33	32.12 ± 0.14	21.31 ± 0.14

Table 6. Accuracy on SOP

	Concatenated (512-dim)			Separated (128-dim)		
	P@1	**RP**	**MAP@R**	**P@1**	**RP**	**MAP@R**
Pretrained	50.71	25.97	23.44	47.25	23.84	21.36
Contrastive	73.12 ± 0.20	47.29 ± 0.24	44.39 ± 0.24	69.34 ± 0.26	43.41 ± 0.28	40.37 ± 0.28
Triplet	72.65 ± 0.28	46.46 ± 0.38	43.37 ± 0.37	67.33 ± 0.34	40.94 ± 0.39	37.70 ± 0.38
NT-Xent	74.22 ± 0.22	48.35 ± 0.26	45.31 ± 0.25	69.88 ± 0.19	43.51 ± 0.21	40.31 ± 0.20
ProxyNCA	75.89 ± 0.17	50.10 ± 0.22	47.22 ± 0.21	71.30 ± 0.20	44.71 ± 0.21	41.74 ± 0.21
Margin	70.99 ± 0.36	44.94 ± 0.43	41.82 ± 0.43	65.78 ± 0.34	39.71 ± 0.40	36.47 ± 0.39
Margin / class	72.36 ± 0.30	46.41 ± 0.40	43.32 ± 0.41	67.56 ± 0.42	41.37 ± 0.48	38.15 ± 0.49
N. Softmax	75.67 ± 0.17	50.01 ± 0.22	47.13 ± 0.22	**71.65 ± 0.14**	**45.32 ± 0.17**	**42.35 ± 0.16**
CosFace	75.79 ± 0.14	49.77 ± 0.19	46.92 ± 0.19	70.71 ± 0.19	43.56 ± 0.21	40.69 ± 0.21
ArcFace	**76.20 ± 0.27**	**50.27 ± 0.38**	**47.41 ± 0.40**	70.88 ± 1.51	44.00 ± 1.26	41.11 ± 1.22
FastAP	72.59 ± 0.26	46.60 ± 0.29	43.57 ± 0.28	68.13 ± 0.25	42.06 ± 0.25	38.88 ± 0.25
SNR	73.40 ± 0.09	47.43 ± 0.13	44.54 ± 0.13	69.45 ± 0.10	43.34 ± 0.12	40.31 ± 0.12
MS	74.50 ± 0.24	48.77 ± 0.32	45.79 ± 0.32	70.43 ± 0.33	44.25 ± 0.38	41.15 ± 0.38
MS+Miner	75.09 ± 0.17	49.51 ± 0.20	46.55 ± 0.20	71.25 ± 0.15	45.19 ± 0.16	42.10 ± 0.16
SoftTriple	76.12 ± 0.17	50.21 ± 0.18	47.35 ± 0.19	70.88 ± 0.20	43.83 ± 0.20	40.92 ± 0.20

Table 5. Accuracy on Cars196

	Concatenated (512-dim)			Separated (128-dim)		
	P@1	**RP**	**MAP@R**	**P@1**	**RP**	**MAP@R**
Pretrained	46.89	13.77	5.91	43.27	13.37	5.64
Contrastive	81.78 ± 0.43	35.11 ± 0.45	24.89 ± 0.50	69.80 ± 0.38	27.78 ± 0.34	17.24 ± 0.35
Triplet	79.13 ± 0.42	33.71 ± 0.45	23.02 ± 0.51	65.68 ± 0.58	26.67 ± 0.36	15.82 ± 0.36
NT-Xent	80.99 ± 0.54	34.96 ± 0.38	24.40 ± 0.41	68.16 ± 0.36	27.66 ± 0.23	16.78 ± 0.24
ProxyNCA	83.56 ± 0.27	35.62 ± 0.28	25.38 ± 0.31	73.46 ± 0.23	28.90 ± 0.22	18.29 ± 0.22
Margin	81.16 ± 0.50	34.82 ± 0.31	24.21 ± 0.34	68.24 ± 0.35	27.25 ± 0.19	16.40 ± 0.20
Margin / class	80.04 ± 0.61	33.78 ± 0.51	23.11 ± 0.55	67.54 ± 0.60	26.68 ± 0.40	15.88 ± 0.39
N. Softmax	83.16 ± 0.25	36.20 ± 0.26	26.00 ± 0.30	72.55 ± 0.18	29.35 ± 0.20	18.73 ± 0.20
CosFace	**85.52 ± 0.24**	37.32 ± 0.28	27.57 ± 0.30	**74.67 ± 0.20**	29.01 ± 0.11	18.80 ± 0.12
ArcFace	85.44 ± 0.28	37.02 ± 0.29	27.22 ± 0.30	72.10 ± 0.37	27.29 ± 0.17	17.11 ± 0.18
FastAP	78.45 ± 0.52	33.61 ± 0.54	23.14 ± 0.56	65.08 ± 0.36	26.59 ± 0.36	15.94 ± 0.34
SNR	82.02 ± 0.48	35.22 ± 0.43	25.03 ± 0.48	69.69 ± 0.46	27.55 ± 0.25	17.13 ± 0.26
MS	85.14 ± 0.29	**38.09 ± 0.19**	**28.07 ± 0.22**	73.77 ± 0.19	**29.92 ± 0.16**	**19.32 ± 0.18**
MS+Miner	83.67 ± 0.34	37.08 ± 0.31	27.01 ± 0.35	71.80 ± 0.22	29.44 ± 0.21	18.86 ± 0.20
SoftTriple	84.49 ± 0.26	37.03 ± 0.21	27.08 ± 0.21	73.69 ± 0.21	29.29 ± 0.16	18.89 ± 0.16

Circle Loss: A Unified Perspective of Pair Similarity Optimization

Yifan Sun[1]*, Changmao Cheng[1]*, Yuhan Zhang[2]*, Chi Zhang[1], Liang Zheng[3], Zhongdao Wang[4], Yichen Wei[1]†
[1]MEGVII Technology [2]Beihang University [3]Australian National University [4]Tsinghua University
{peter, chengchangmao, zhangchi, weiyichen}@megvii.com

Abstract

This paper provides a pair similarity optimization viewpoint on deep feature learning, aiming to maximize the within-class similarity s_p and minimize the between-class similarity s_n. We find a majority of loss functions, including the triplet loss and the softmax cross-entropy loss, embed s_n and s_p into similarity pairs and seek to reduce $(s_n - s_p)$. Such an optimization manner is inflexible, because the penalty strength on every single similarity score is restricted to be equal. Our intuition is that if a similarity score deviates far from the optimum, it should be emphasized. To this end, we simply re-weight each similarity to highlight the less-optimized similarity scores. It results in a Circle loss, which is named due to its circular decision boundary. The Circle loss has a unified formula for two elemental deep feature learning paradigms, i.e., learning with class-level labels and pair-wise labels. Analytically, we show that the Circle loss offers a more flexible optimization approach towards a more definite convergence target, compared with the loss functions optimizing $(s_n - s_p)$. Experimentally, we demonstrate the superiority of the Circle loss on a variety of deep feature learning tasks. On face recognition, person re-identification, as well as several fine-grained image retrieval datasets, the achieved performance is on par with the state of the art.

1. Introduction

This paper holds a similarity optimization view towards two elemental deep feature learning paradigms, *i.e.*, learning from data with class-level labels and from data with pair-wise labels. The former employs a classification loss function (*e.g.*, softmax cross-entropy loss [25, 16, 36]) to optimize the similarity between samples and weight vectors. The latter leverages a metric loss function (*e.g.*, triplet loss [9, 22]) to optimize the similarity between samples. In our interpretation, there is no intrinsic difference between these two learning approaches. They both seek to minimize between-class similarity s_n, as well as to maximize within-class similarity s_p.

From this viewpoint, we find that many popular loss functions (*e.g.*, triplet loss [9, 22], softmax cross-entropy loss and its variants [25, 16, 36, 29, 32, 2]) share a similar optimization pattern. They all embed s_n and s_p into similarity pairs and seek to reduce $(s_n - s_p)$. In $(s_n - s_p)$, increasing s_p is equivalent to reducing s_n. We argue that this symmetric optimization manner is prone to the following two problems.

• **Lack of flexibility for optimization.** The penalty strength on s_n and s_p is restricted to be equal. Given the specified loss functions, the gradients with respect to s_n and s_p are of same amplitudes (as detailed in Section 2). In some corner cases, *e.g.*, s_p is small and s_n already approaches 0 ("A" in Fig. 1 (a)), it keeps on penalizing s_n with a large gradient. It is inefficient and irrational.

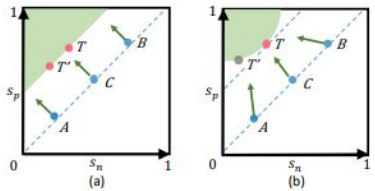


Figure 1: Comparison between the popular optimization manner of reducing $(s_n - s_p)$ and the proposed optimization manner of reducing $(\alpha_n s_n - \alpha_p s_p)$. (a) Reducing $(s_n - s_p)$ is prone to inflexible optimization (A, B and C all have equal gradients with respect to s_n and s_p), as well as ambiguous convergence status (both T and T' on the decision boundary are acceptable). (b) With $(\alpha_n s_n - \alpha_p s_p)$, the Circle loss dynamically adjusts its gradients on s_p and s_n, and thus benefits from a flexible optimization process. For A, it emphasizes on increasing s_p; for B, it emphasizes on reducing s_n. Moreover, it favors a specified point T on the circular decision boundary for convergence, setting up a definite convergence target.

*Equal contribution.
†Corresponding author.

1. Cosface loss
2. Arcface loss
3. Circle loss