MaskCon: Masked Contrastive Learning for Coarse-Labelled Dataset

Chen Feng, Ioannis Patras
Queen Mary University of London, UK
{chen.feng, i.patras}@qmul.ac.uk

A: Implementation and dataset details

Implementation details

Hyperparameter settings For datasets except ImageNet-1K, we use the SGD optimizer with a momentum of 0.9, and train for 200 epochs with a batch size of 128, a learning rate of 0.02 and cosine annealing lr scheduler, and weight decay as 0.0005. For ImageNet-1K dataset, we train for 100 epochs with a batch size of 256. For contrastive learning, we use a memory bank with 8,192 elements, an MLP projector with a hidden dimension of 512, an output dimension of 128, and a temperature for the projection head as $\tau_0 = 0.1$. Moreover, we show the MaskCon-specific hyperparameters τ and w corresponding to the reported numbers in the main paper below (Tab. 1). This can be further validated in APPENDIX B.

Dataset	SOP-split1		SOP-split2		Cars196	
	w	τ	w	τ	w	τ
Grafit	0.8	/	0.8	/	1	/
CoIns	0.5	/	0.2	/	1	/
Ours	0.8	0.1	0.5	0.05	1	0.1

Table 1. Hyperparameters in the main paper.

Augmentation strategies The augmentation strategy is critical for contrastive learning, especially for self-supervised contrastive learning. Following [2,4], we apply strong and weak augmentations for the query and key view respectively. As discussed in the main paper, the strong augmentation may be destructive for fine-grained image classification. In particular, considering that each class in the SOP dataset consists of different views of the same product, we, therefore, use weak image augmentations for both query and key view. We list the applied data augmentations for each dataset as below in the PYTORCH format, the complete repo will be released upon acceptance.

Strong augmentation

```
if dataset == 'cars196':
    strong_transform = transforms.Compose([
    transforms.RandomResizedCrop(size, (0.2, 1)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomPerspective(0.5, 0.5),
    transforms.RandomApply([transforms.
    ColorJitter(0.4, 0.4, 0.4, 0.1)], p=0.8),
    transforms.RandomGrayscale(p=0.2),
    transforms.RandomApply(
    [GaussianBlur([.1, 2.])], p=0.5),
    transforms.ToTensor(),
    normalize,
    ])
elif dataset == 'sop_split1' or dataset == '
    sop_split2':
    strong_transform = transforms.Compose([
    transforms.RandomResizedCrop(size, (0.2, 1)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomPerspective(0.5, 0.5),
    transforms.ToTensor(),
    normalize,
    ])
else:
    strong_transform = transforms.Compose([
    transforms.RandomResizedCrop(size, (0.2, 1)),
    transforms.RandomHorizontalFlip(),
    CIFAR10Policy(),
    transforms.ToTensor(),
    normalize,
    ])
```

Weak augmentation

```
if dataset == 'cars196' or dataset == 'sop_split1
    ' or dataset == 'sop_split2':
    weak_transform = transforms.Compose([
    transforms.RandomResizedCrop(size, (0.2, 1)),
    transforms.RandomPerspective(0.5, 0.5),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    normalize
    ])
else:
    weak_transform = transforms.Compose([
    transforms.RandomResizedCrop(size, (0.2, 1)),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    normalize,
    ])
```

Test augmentation

```
test_transform = transforms.Compose([
    transforms.Resize(size),
    transforms.ToTensor(),
    normalize,
    ])
```

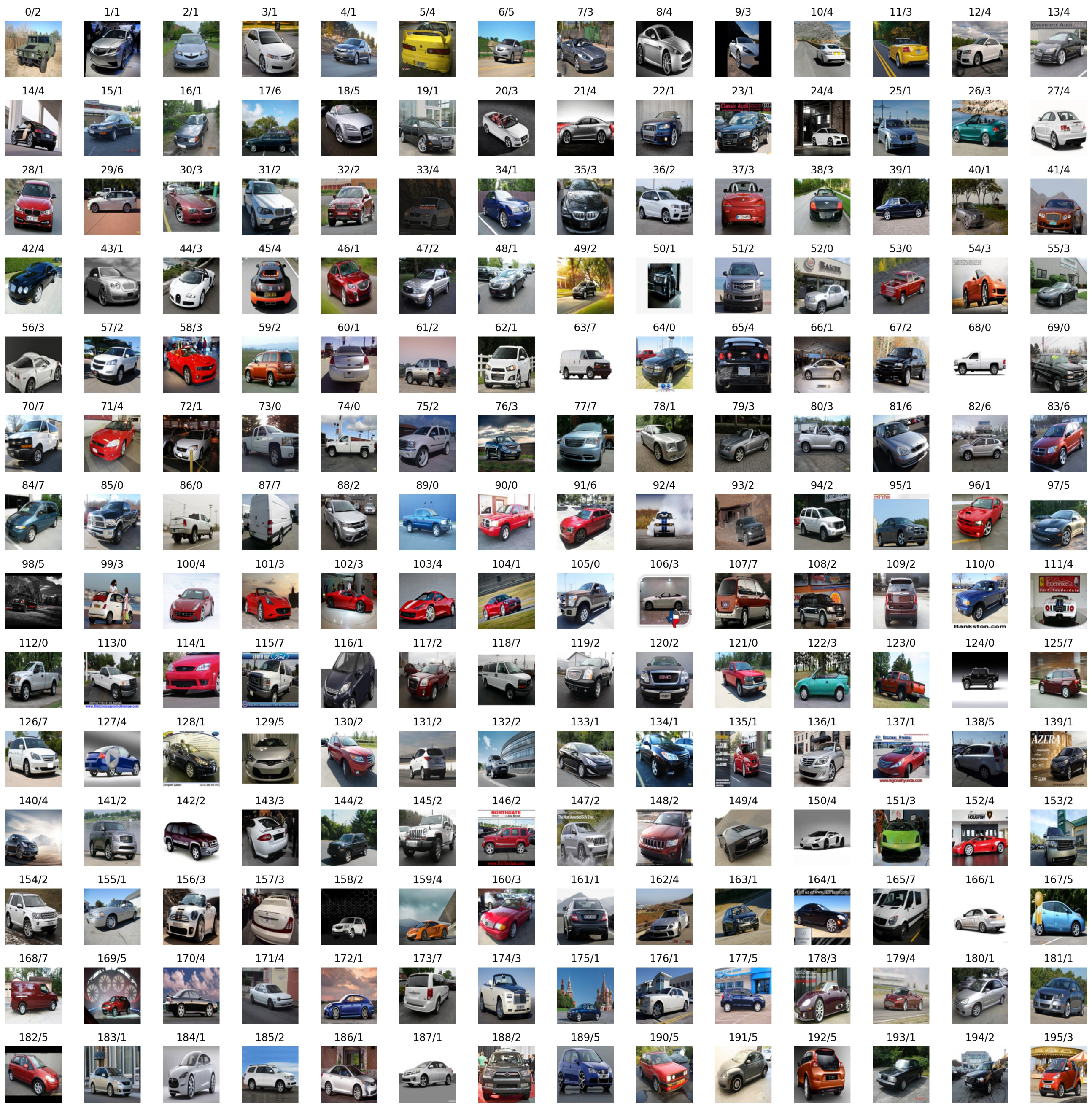

Figure 1. Coarse labels for each of the fine classes—each image captioned as {fine label}/{coarse label}.

Dataset details

Stanford Online Products (SOP) For Stanford Online Products (SOP) datasets, we artificially generate two different splits. There are very limited images for each product in the original SOP datasets, ranging from 2 to 12 (Fig. 2). Considering an extreme case, that is, for each product there is only one view image, which will lead us to self-supervised learning based on instance discrimination. To avoid this, we thus select those classes with more images. More specifically, for SOP-split1, we select the classes with eight or more images and obtain a subset consisting of 5,035 classes. We evenly split (2,517 train classes and 2,518 test classes) this subset to obtain the training set and the test set with 25,368 and 25,278 images, respectively. For SOP-split2, we select all products with twelve images, and then randomly select ten train images and two test images. Thus, we have 1,498 classes with a total of 17,976 images (2,996 test images and 14,980 train images), respectively.

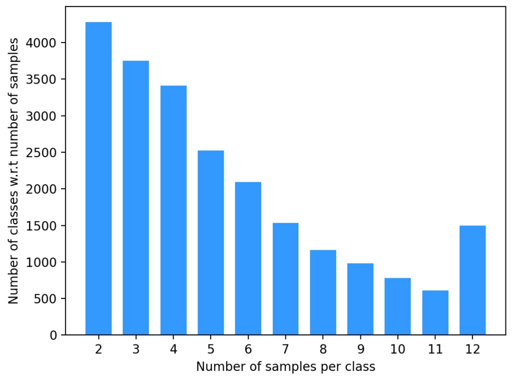

Figure 2. Statistics on SOP dataset.

Stanford Cars196 For the Cars196 dataset, similarly, there are no official coarse labels. For this reason, we manually group 196 car models into 8 coarse classes based on the common types of cars: '0: Cab', '1: Sedan', '2: SUV', '3: Convertible', '4: Coupe', '5: Hatchback', '6: Wagon' and '7: Van'. In Fig. 1, we provide some example images

0	1	2	3	4	5	6	7
18	50	36	30	29	14	7	12

Table 2. Number of fine classes in each coarse class on Cars196.

from original fine classes to coarse classes for the Stanford Cars196 dataset.

ImageNet-1K In this section, we evaluate our method on the large-scale ImageNet-1K dataset. For efficiency, we experiment with the downsampled version of ImageNet-1K dataset [1], with each sample resized to 32×32. Since there

0	1	2	3	4	5	6	7	8	9	10	11
61	123	59	95	60	130	70	53	105	71	93	80

Table 3. Number of fine classes in each coarse class.

are no officially coarse labels for imagenet, we here introduce coarse labels based on the WordNet [3] hierarchy so as to artificially group the whole dataset into 12 coarse classes: '0: Invertebrate', '1: Domestic animal', '2: Bird', '3: Mammal', '4: Reptile/Aquatic vertebrate', '5: Device', '6: Vehicle', '7: Container', '8: Instrument', '9: Artifact', '10: Clothing' and '11: Others'.

Class imbalance We note that the number of fine classes in each coarse class is highly imbalanced (Tab. 3, Tab. 2). In this work, we do not explicitly deal with it (for e.g., dataset resampling), as we believe, that such class imbalance can be common in the targeted problem setting, and we aim to test our method in such more realistic setting.

B: Additional experiments and results

As mentioned in the main content, we perform a search for the hyperparameters for both Grafit and CoIns and report the best results. Here, in Tab. 4, Tab. 5, Tab. 7, Tab. 6 and Tab. 8, we report the results with so additional different hyperparameter settings.

Method	w	τ	Recall@1	Recall@2	Recall@5	Recall@10
SupFINE	/	/	71.13	80.03	87.61	91.59
Grafit	0.2	/	47.80	59.41	73.29	82.17
	0.5	/	57.78	68.24	80.01	87.05
	0.8	/	**60.57**	**71.13**	**82.32**	**89.21**
	1.0	/	58.65	70.04	82.18	89.09
CoIns	0.2	/	56.93	68.51	80.90	87.88
	0.5	/	**60.10**	**70.89**	**83.14**	**89.52**
	0.8	/	59.35	70.40	83.19	89.82
	1.0	/	47.25	61.24	77.78	87.00
MaskCon	0.5	0.01	54.22	65.31	77.45	85.12
	0.5	0.05	58.89	68.91	80.02	86.89
	0.5	0.1	60.18	70.41	81.81	88.25
	0.5	0.5	57.46	68.27	79.86	87.05
	1	0.01	61.41	71.22	81.44	87.70
	1	0.05	**65.52**	**74.46**	**83.64**	**89.25**
	1	0.1	62.52	72.51	83.27	89.18
	1	0.5	59.76	70.28	81.97	88.58
SelfCon	/	/	40.50	51.83	66.23	76.66

Table 4. Extra results on CIFAR100 dataset.

Method	w	τ	Recall@1	Recall@2	Recall@5	Recall@10
SupFINE	/	/	83.94	88.04	91.95	94.00
Grafit	0.2	/	71.04	76.17	81.85	85.51
	0.5	/	72.66	77.43	82.89	86.40
	0.8	/	**74.02**	**78.82**	**84.13**	**87.91**
	1.0	/	53.69	59.55	67.12	72.78
CoIns	0.2	/	70.75	76.32	82.24	86.16
	0.5	/	**70.84**	**76.01**	**82.20**	**86.08**
	0.8	/	67.68	73.26	80.04	84.18
	1.0	/	36.35	42.39	50.30	56.52
MaskCon	0.2	0.1	72.35	77.34	82.92	86.46
	0.5	0.05	70.87	76.17	82.11	85.92
	0.5	0.1	73.22	78.34	83.98	87.35
	0.8	0.05	68.53	73.72	79.86	84.22
	0.8	0.1	**74.05**	**78.97**	**84.48**	**87.96**
SelfCon	/	/	70.36	75.57	81.53	85.13

Table 5. Extra results on SOP-split1 dataset.

Please note that we do not investigate fewer values for the hyperparameter w in comparison to Grafit and CoIns. From the tables we can see that with appropriate weights, both Grafit and CoIns get considerable improvements over the supervised and self-supervised only baselines. Our

Method	w	τ	CIFARtoy (good split)				CIFARtoy (bad split)			
			Recall@1	Recall@2	Recall@5	Recall@10	Recall@1	Recall@2	Recall@5	Recall@10
SupFINE	/	/	94.14	96.61	98.03	98.65	94.11	96.53	98.45	98.96
Grafit	0.2	/	85.34	91.38	96.05	97.85	86.33	92.31	96.63	98.18
	0.5	/	**86.61**	**92.33**	**97.01**	**98.38**	87.94	93.63	97.59	98.76
	0.8	/	86.23	92.56	96.69	98.50	**89.96**	**94.36**	**97.71**	**98.90**
	1.0	/	73.84	84.25	92.14	95.46	84.66	90.93	95.15	96.71
CoIns	0.2	/	85.74	91.88	96.41	98.60	89.80	94.13	97.18	98.30
	0.5	/	86.00	92.25	96.91	98.60	90.14	94.11	97.78	98.70
	0.8	/	**86.15**	**92.76**	**97.21**	**98.76**	**90.55**	**94.94**	**97.73**	**98.71**
	1.0	/	76.30	8.26	94.65	97.46	87.15	92.85	96.78	98.34
MaskCon	0.5	0.01	87.44	93.01	96.95	98.38	88.89	93.84	97.20	98.38
	0.5	0.05	88.09	93.18	96.99	98.51	90.21	94.41	97.41	98.56
	0.5	0.1	87.13	92.95	96.93	98.44	88.29	93.61	97.25	98.54
	0.5	0.5	86.31	92.49	96.74	98.53	88.46	93.43	97.63	98.70
	1	0.01	88.24	93.44	97.01	98.26	90.53	94.23	97.51	98.55
	1	0.05	**90.28**	**94.04**	**97.33**	**98.53**	**91.56**	**95.23**	**97.70**	**98.70**
	1	0.1	78.25	87.10	94.26	96.85	88.44	93.65	97.10	98.34
	1	0.5	79.58	88.10	94.58	96.99	88.25	93.33	96.99	98.19
SelfCon	/	/	84.84	91.73	96.14	97.94	84.83	91.55	96.35	98.16

Table 6. Extra results on CIFARtoy dataset.

Method	w	τ	Recall@1	Recall@2	Recall@5	Recall@10
SupFINE	/	/	69.56	75.70	83.24	87.88
Grafit	0.2	/	36.85	42.52	50.23	57.04
	0.5	/	37.98	43.69	51.67	57.98
	0.8	/	**39.12**	**44.66**	**53.10**	**59.65**
	1.0	/	25.07	29.24	35.85	41.59
CoIns	0.2	/	**38.22**	**45.19**	**54.37**	**61.18**
	0.5	/	38.18	45.06	54.17	61.28
	0.8	/	36.72	42.49	51.23	58.61
	1.0	/	22.56	26.34	33.28	38.95
MaskCon	0.2	0.05	39.69	45.69	53.70	60.05
	0.5	0.05	**45.36**	**51.07**	**58.91**	**65.52**
	0.8	0.05	43.84	50.03	58.18	64.19
	1.0	0.05	41.05	46.70	54.74	60.98
SelfCon	/	/	35.85	41.46	49.77	56.11

Table 7. Extra results on SOP-split2 dataset.

Method	w	τ	Recall@1	Recall@2	Recall@5	Recall@10
SupFINE	/	/	78.09	82.97	86.51	88.04
Grafit	0.2	/	23.64	32.69	46.35	57.98
	0.5	/	28.83	38.96	53.99	65.08
	0.8	/	35.58	47.97	64.47	75.28
	1.0	/	**42.30**	**54.79**	**71.10**	**81.74**
CoIns	0.2	/	32.30	43.19	59.11	71.10
	0.5	/	36.71	48.80	65.33	76.16
	0.8	/	38.71	49.04	67.28	79.23
	1.0	/	**42.77**	**55.60**	**72.29**	**82.53**
MaskCon	1	0.01	27.87	38.19	52.80	64.13
	1	0.05	33.04	43.69	59.10	70.59
	1	0.1	**45.53**	**58.56**	**74.36**	**84.36**
	1	0.5	42.84	55.23	71.57	81.72
SelfCon	/	/	20.97	28.75	41.44	52.61

Table 8. Extra results on Stanford Cars196 dataset.

method, achieves even higher improvements with an appropriate τ.

References

[1] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. *arXiv preprint arXiv:1707.08819*, 2017. 3

[2] Chen Feng and Ioannis Patras. Adaptive soft contrastive learning. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 2721–2727, 2022. 1

[3] George A Miller. *WordNet: An electronic lexical database*. MIT press, 1998. 3

[4] Mingkai Zheng, Shan You, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Ressl: Relational self-supervised learning with weak augmentation. *Advances in Neural Information Processing Systems*, 34:2543–2555, 2021. 1