Clever Chimpanzee: Detection and Identification for Chimpanzees

Chuyue Tang
Yuanpei College
Peking University
2000017814@stu.pku.edu.cn

Siwen Xie
Yuanpei College
Peking University
swxie@stu.pku.edu.cn

Abstract

Human face recognition systems have walked into everyone's life, enabling automatic detection and classification of different people's faces. However, few have noticed that as a common laboratory animal for biology and cognitive science, chimpanzees also need such a system. In this work, we propose a model for chimpanzee detection and identification on a limited and imbalanced dataset. For detection, we perform finetuning on pre-trained detection models and adjust the NMS strategy to mitigate ambiguity, achieving about 0.89 mAP. As for classification, we try to increase its classification ability and solve the problem of data imbalance by 1) supervised learning with large margin cosine loss; 2) supervised contrastive learning. 1) is first designed for deep human face recognition. We transfer it to our setting and also achieve a good result. With 2), we compare the effect of self-supervised contrastive learning and supervised contrastive learning. Our methods achieve a SOTA performance of around 77% on the validation set. We also propose a two-stage training in the hope of combing the advantages of the two methods. Our code can be found in https://github.com/tcy63/CV_chimpanzee.

1 Introduction

Chimpanzees are a common laboratory animal for research fields like primatology, biology, and cognitive science. In order to keep track of the primates' behavior, researchers set up cameras in their natural or artificial habitats. However, the obtained video data is usually underexploited, due to the unbearable amount of time and effort required to manually check and analyze the videos. This difficulty can be mitigated by using an automatic visual system, which is proven possible by existing approaches to tracking humans [4] and other animals [10] from video clips.

A visual model for chimpanzees will have to tackle three fundamental tasks: (i) where they are (detection), (ii) who they are (identification), and (iii) what they are doing (2D pose estimation). Detection requires the model to take a frame of the video as input, and predict bounding boxes for all chimpanzee entities in the input frame, which is the basis of the other two tasks. Identification for chimpanzees is a classification task, where the input is an image of a single chimpanzee (cropped according to its bounding box), and the label is its name. 2D pose estimation takes the same input as identification, but the model needs to predict the key points (or skeleton) of the chimpanzee's body instead of its name. In this paper, we only attempt the former two tasks, leaving out, with regret, the pose estimation task due to limited time.

Detecting and identifying chimpanzees are not easy tasks. They share similar challenges with the better-studied task of pedestrian detection and re-identification [15], namely occlusion, angle variance, pose variance, and illumination change. Despite such similarities, chimpanzees have fewer expressive features compared with humans. For example, the clothing of humans can serve as a distinctive clue for detection and identification, but chimpanzees are covered with dark fur, which makes them not only indistinguishable from each other but also hard to separate from the background. Moreover,

unlike for humans, datasets for chimpanzees can be very limited and strongly imbalanced (see Section 3), adding even more difficulties to the task.

In this paper, we propose separate models for the two tasks, achieving 0.889 mAP for detection and 77.42% accuracy for identification, prospectively facilitating chimpanzee data analysis in a wide range of research fields. Details of the models will be illustrated in the following sections. Section 2 reviews previous work related to our research. Section 3 introduces the chimpanzee dataset. Section 4 presents our methods and experimental results for detection, and Section 5 presents that for identification. Finally, our conclusion and ideas about future work will be presented in Section 6.

2 Related work

Object detection Object detection is a well-known and well-studied problem in computer vision. It aims at predicting the bounding boxes and labels for all objects in the input image. YOLO [11] is an end-to-end architecture for object detection, which yields remarkable performance in a relatively short runtime. YOLOv5 [1] is the up-to-date version of YOLO, which offers pre-trained models of various scales (*e.g.*, `yolov5s`, `yolov5m`, `yolov5x`, from small to large), neatly-organized code and convenient APIs, which is why it was selected as our codebase. The dataset YOLOv5 used for pretraining and testing was COCO [8]. COCO is a popular detection dataset, covering objects of 91 classes, but does not include data for chimpanzees. Therefore, we made extra efforts to adapt YOLOv5 models to chimpanzee detection (see Section 4).

Human face recognition Compared with little research in chimpanzee recognition, human face recognition is one of the oldest and most important computer vision applications. It has gone through the stage where handcraft or specially learned features are used and has shifted to deep learning methods since Facebook's DeepFace [12] achieved SOTA performance on Labeled Faces in the Wild (LFW) in 2014. Progress has been made by refining both the deep neural network structures and the loss functions. The architectures can be categorized as backbone which includes many different typical CNN architectures and assembled networks. As for the loss function, many works focus on improving the common softmax loss in order to make more discriminative features. CosFace [13] is one of them, which makes learned features separable with larger cosine distance. Since their works and ours share the same goal of reducing intra-class variance and enlarging inter-class variance.

Few-shot learning The few-shot learning paradigm is proposed in order to learn from a limited number of examples with supervised information [14]. In the classification task, one usually considers N-way-K-shot classification. Methods like transfer learning and meta-learning have been found successful at solving few-shot classification problems.

Contrastive learning Contrastive learning aims at learning low-dimensional representations of the data that contrast similar and dissimilar samples. Intuitively, it pulls features of the same class closer to each other and pushes dissimilar ones apart in a feature space. Many objectives have been developed, such as contrastive loss, triplet loss, N-pair loss, and NCE loss. This perspective may be helpful if we would like to make more fine-grained representations of chimpanzees.

Learning from imbalanced data The imbalanced learning problem occurs when the training dataset has skewed class proportions. It has many real-world motivations such as rare event detection. In the past decades, methods have been focused on re-sampling, re-weighting, or some domain-specific techniques. More recently, [2] propose a theoretically-principled label-distribution-aware margin (LDAM) loss motivated by minimizing a margin-based generalization bound, which gives minority classes with larger margins. In contrast to other margin losses that are class-independent, they theoretically prove that uneven margins achieve good empirical progress.

3 The chimpanzee dataset

3.1 Basic information

All of the data comes from Leipzig Zoo by Prof. Federico's team.

Detection It is a multi-object detection task. The training set has 1188 images with one or more chimpanzees in different scenes, and the validation set has 94 images.

Identification The identification task is done on the ground truth bounding boxes in the subset of the detection set. It has 662 images for training and 278 images for validation. There are 17 classes in total, but with a skewed distribution.

3.2 Features of the dataset

The detailed information including the number of training and validation samples for each class is in Appendix A.

Few-shot learning Few-shot learning aims to learn from only k samples for each class, where k is usually 1, 5, 10, e.t.c. Although our dataset is not aimed at the study of few-shot learning, many minority classes do meet the standard of few-shot learning. There are 8 out of 17 classes that have less than 10 training samples.

Imbalanced learning [2] create imbalanced CIFAR-10 and CIFAR-100 for experimental study by reducing the number of training examples per class. They use the imbalanced ratio $\rho = 100/10$ as the ratio between the most and least frequent classes. They also evaluate two different types of imbalance, long-tailed and step. The former follows an exponential decay over the numbers of classes and the latter maintains the same size within the minority and majority classes, where the fraction of the minority class is $\mu = 0.5$. However, as shown in the appendix, $\rho_{\text{ours}} > 200$. And our data tends to be distributed in a long-tail way, with the fraction of the minority class $\mu_{\text{ours}} < 0.5$.

The above observation and comparison suggest that our dataset is intrinsically harder than relevant works. On the one hand, we may take inspiration from previous methods. On the other hand, the uniqueness of our dataset motivates us to re-design these methods to serve our purpose.

4 Detection

4.1 Baseline

For the detection problem, we chose YOLOv5 as the codebase for its convenience and acknowledgeable performance. We directly test 2 pre-trained detection models of different sizes, `yolov5s` and `yolov5x`, on our validation set. Since these models were pre-trained on COCO, which did not include chimpanzees, we set the model to single-class mode, taking all its valid detections as predictions for chimpanzees.

However, the visualized results showed some false positives, such as the zoo facilities. Therefore, it would be more reasonable to integrate a classification module that can rule out irrelevant objects. We chose YOLO's classification model yolov5x-cls, pre-trained on ImageNet which included chimpanzee data, and fed the detection results into this model. A detection would be accepted only if "chimpanzee" was in its top-5 classification labels. Test results with and without this module are presented in Table 1, from which we can see that the larger-sized pre-trained model performs significantly better. The classification module did not make much difference. We assume this is because the classification module is not strong enough to classify all chimpanzees correctly, and discards some true positives. We plan to conduct more experiments for further validation of this assumption.

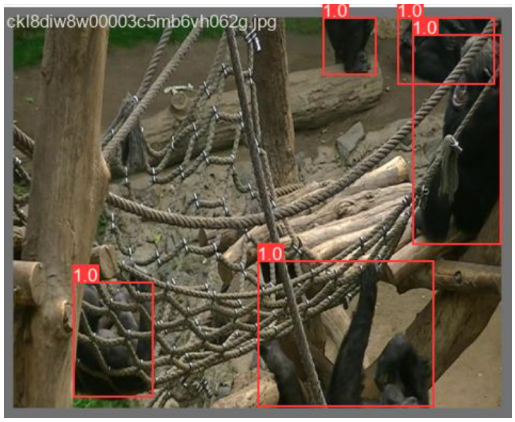
4.2 Finetuning

To improve performance, we finetuned the pre-trained models on our training set, setting epochs=100, batch_size=16, and selecting the best model during validation. Results are shown in Table 1. Finetuning significantly improves the models' overall performance, reaching an mAP of about 0.88.

Besides vanilla finetuning, we also tried finetuning with frozen layers, for the purpose of speeding up the training process and keeping some of the models' general knowledge. We experimented on `yolov5s` and `yolov5x`, freezing the first 0 (freeze none), 4 (freeze the first module), or 10 (freeze the backbone and leave out the head) layers. From the results shown in Table 1, we can see that freezing layers allows the model to reach a comparable result in less training time, although its performance does decline when more layers are frozen.

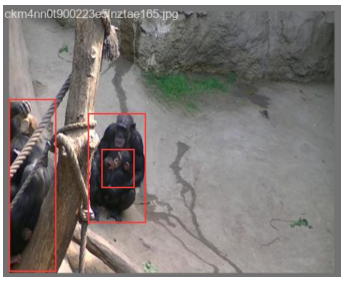
We visualized the validation results to see what else could be improved and discovered that (i) the labels in the original dataset were not perfect, but our model managed to detect some chimpanzees that were not annotated in the ground truth labels (see an example in Figure 1), showing remarkable

(a) Label

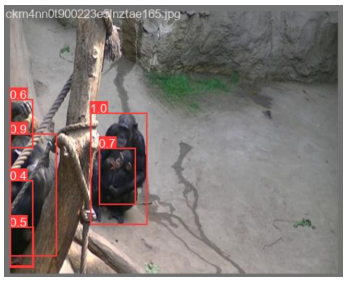


(b) Our model's prediction

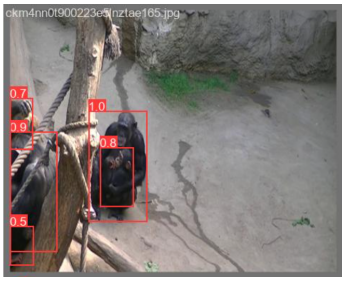
Figure 1: This example shows the robustness of our model. Two chimpanzees on the bottom were not annotated, but our model made the accurate detection with high confidence.



(a) Label



(b) Prediction with vanilla NMS



(c) Prediction with Softer-NMS

Figure 2: A typical failure case of our model. The chimpanzee on the left is occluded by ropes and branches, making it hard to detect. In such cases, our model tends to predict multiple bounding boxes for a single chimpanzee. This issue is slightly mitigated by using Softer-NMS.

robustness, and (ii) a majority of the failure cases were of crowded scenes, where our model tends to propose multiple bounding boxes for a single chimpanzee, especially when it is occluded (see an example in Figure 2).

yolov5s	yolov5x	cls module	finetune frozen layers	P	R	mAP50	mAP50-95
✓			–	0.592	0.394	0.470	0.292
✓		✓	–	0.592	0.393	0.470	0.290
	✓		–	0.680	0.483	0.577	0.379
	✓	✓	–	0.678	0.480	0.576	0.376
✓			0	0.882	0.799	0.882	0.694
✓			4	**0.924**	0.775	0.881	0.689
✓			10	0.892	0.769	0.863	0.628
	✓		0	0.887	**0.839**	**0.884**	**0.735**
	✓		4	0.895	0.818	0.882	0.721
	✓		10	0.904	0.806	0.884	0.714

Table 1: Results for detection baseline and finetuning.

4.3 Different strategies for NMS

To address the above issue, we tried out various strategies for Non-Maximum Suppression (NMS), as it serves for sifting out the true positives from overlapping bounding boxes. The original implementation of YOLOv5 used the vanilla NMS, which removes lower-scoring boxes that have an IoU greater than a threshold with another higher-scoring box. A primitive idea is to adjust the hyperparameter IoU

threshold, and we experimented on this using the model finetuned on `yolov5x` with 4 frozen layers. The results are shown in Table 2. We can see that 0.5 serves as a relatively appropriate threshold.

We also tried another NMS method, Softer-NMS [5] (or Merge-NMS), which might be helpful in crowded scenes. Softer-NMS computes the weighted mean of overlapping boxes according to their scores, merging them to generate the final prediction. We implemented Softer-NMS and experimented on different IoU thresholds, and found that Softer-NMS is more sensitive to a lower threshold. The visualized results in Figure 2 demonstrate that Softer-NMS mitigates the duplication issue of vanilla NMS, but this improvement is not significantly shown in the metrics in Table 2.

We have also learned about more sophisticated methods for NMS, such as using a simple network for NMS [6], and NMS-aware training [9], which we might try in the future.

NMS method	IoU threshold	P	R	mAP50	mAP50-95
vanilla	0.4	**0.903**	0.818	0.886	0.730
vanilla	0.5	0.901	0.818	**0.889**	0.729
vanilla	0.55	0.901	0.818	0.887	0.725
softer	0.4	0.877	**0.844**	0.885	**0.732**
softer	0.5	0.901	0.818	**0.889**	**0.732**
softer	0.55	0.901	0.818	0.884	0.729

Table 2: Results for different NMS strategies.

5 Identification

5.1 Supervised learning with large margin cosine loss (LMCL)

5.1.1 Large margin cosine loss

Large Margin Cosine Loss was first proposed by Wang et al. [13]. It aims to learn discriminative features by maximizing inter-class cosine margin. Our work first adopts this loss function in chimpanzee recognition and gets relatively good results.

The weight features W are trained together with the backbone model during training. This loss is formalized as follows:

$$L_{lmc} = \frac{1}{N}\sum_{i=1}^{N} -\log \frac{e^{s(\cos(\theta_{y_i},i)-m)}}{e^{s(\cos(\theta_{y_i},i)-m)} + \sum_{j\neq y_i} e^{s(\cos(\theta_j,i))}}$$

$$\cos(\theta_j, i) = W_j^T \cdot x_i, W_j = \frac{W_j^*}{||W_j^*||}, x_i = \frac{x_i^*}{||x_i^*||}$$

This loss has two hyperparameters: the forced margin m and the feature norm s.

During testing, the class of largest cosine similarity is the predicted label.

$$\text{pred}_i = \text{argmax}_j(W_j^T \cdot x_i)$$

5.1.2 Models

The model consists of a ResNet encoder and a linear layer as learnable class features. The output of the model is interpreted as cosine similarity between a new image feature and each class feature embedded in the last linear layer. During training, the output can be fed into the large margin cosine loss. During inference, the class of the largest similarity is taken as the prediction.

5.1.3 Experiments

We set batch size $B = 256$, image size $I = 32$. Single GPU. We tried different optimizers, schedulers, and hyper-parameters of them. And we finally choose Adam with default hyperparameters as our optimizer and ExponentialLR with $\gamma = 0.999$ as our learning rate scheduler.

For all settings, we train the model with 500 epochs, although many of them have converged before the training ends. And we report the best validation accuracy within this stage.

First, we experiment with the ResNet encoder:

- whether to use the pre-trained ResNet encoder from PyTorch;
- the choice of ResNet encoder (ResNet18, ResNet50, ResNet101, ResNet152);

As can be seen from Tab. 3, pre-trained ResNet can provide better feature initialization and achieve consistently better results. As for the encoder's size, we do not observe an obvious advantage of using models smaller or larger than ResNet50.

	ResNet18	ResNet34	ResNet50	ResNet101	ResNet152
w/	**75.63**	**74.91**	**77.42**	**74.55**	**75.63**
w/o	74.91	72.04	73.48	69.53	68.10

Table 3: Best validation accuracy (%) with different encoder structures and the effect of using PyTorch pre-trained models.

Next, we experiment with the hyper-parameters of the large margin cosine loss:

- the choice of margin m;
- whether to use learnable scale s;

In Tab. 4, a smaller margin produces better results when the scalar s is fixed. We suggest that too large a margin may be overly strict for our dataset. The strategy of learnable scalar also produces good results. It significantly increases the performance especially when the margin is large.

	m=0	m=0.1	m=0.2	m=0.3	m=0.4	m=0.5
fixed s	76.62	75.18	**77.42**	71.94	73.02	70.50
learnable s	74.91	74.91	76.98	**77.34**	**76.62**	**75.18**

Table 4: Best validation accuracy (%) with different m and s.

5.2 Supervised contrastive learning

5.2.1 Self-supervised contrastive loss

First, we review a traditional work regarding contrastive learning and introduce some common notations we will use throughout this section. In the self-supervised setting, no label information is available. Each minibatch consists of N samples $\{x_k, y_k\}_{k=1}^{N}$. Then, they will go through three processes:

1. augmentation: each sample is augmented twice, resulting in a multi-viewed batch $\{\tilde{x}_k, \tilde{y}_k\}_{k=1}^{2N}$
2. encoding: each image is encoded by an encoder f (*e.g.* ResNet50), $y = f(x) \in \mathcal{R}^{D_{en}}$
3. projection: each encoding is projected by a projection head g to a low-dimensional feature space, $z = g(y) \in \mathcal{R}^{D_{ft}}$

For each anchor image in the batch of index i, the corresponding augmented image of index $j(i)$ is considered as the positive sample, and the rest are negative samples. Contrastive loss aims to make the projected features of the same class closer to each other, indicated by the cosine similarity $cos(z_i, z_{j(i)}) = \frac{<z_i, z_{j(i)}>}{||z_i|| \cdot ||z_{j(i)}||}$. To stay aligned with the previous notation, we change the temperature scalar τ into $s = \frac{1}{\tau}$.

$$L_{simclr} = \frac{1}{2N} \sum_{i=1}^{2N} -\log \frac{e^{s \cos(z_i, z_{j(i)})}}{\sum_{k \neq i} e^{s \cos(z_i, z_k)}}$$

This is the famous SimCLR loss by Chen et al. [3]

5.2.2 Supervised contrastive loss

Even though the self-supervised contrastive loss is effective when label information is not available, it does not achieve its full potential when we do have all the labels.

Khosla et al. [7] proposes a new contrastive loss by leveraging label information when picking positive and negative samples. It can generalize to an arbitrary number of positive samples in one minibatch.

For anchor image x_i, denote the set of positive examples as $P(i) = \{j \neq i : \tilde{y}_j = \tilde{y}_i\}$.

$$L_{supcon} = \frac{1}{2N} \sum_{i=1}^{2N} -\frac{1}{|P(i)|} \sum_{p \in P(i)} \log \frac{e^{s \cos(z_i, z_p)}}{\sum_{k \neq i} e^{s \cos(z_i, z_k)}}$$

5.2.3 Comparison between self-supervised contrastive loss and supervised contrastive loss

First, both losses preserve the summation over negatives in the contrastive denominator, which is said to increase the discrimination power by adding more negative examples.

Second, the self-supervised contrastive loss is likely to face the problem of 'false negative'. For example, for an imbalanced dataset like ours, you may falsely judge some samples as 'negative'. And the problem becomes more serious as this class makes up largerer portion of the whole dataset.

Minority classes may benefit from this self-supervised setting because false negative is unlikely to happen to them. However, as for majority classes, they suffer more from the false negative problem, and the supervised setting may be more beneficial to them.

Motivated by this observation, we propose a two-stage learning process here. First, we learn a representation that benefits rare classes more by the ordinary self-supervised contrastive loss; Then, we continue to adjust this representation to bring common classes closer by the supervised contrastive loss. We also inspect the inverse process for a clearer understanding.

5.2.4 Models

Training The model consists of a ResNet50 encoder and a two-layer MLP as the projection head.

Inference We can use the trained encoder from the training stage, freeze it, and finetune a linear classifier on top of it as a standard transfer learning practice. Moreover, the classifier can be non-trainable similar to a nearest-neighbor strategy.

5.2.5 Experiments

We set batch size $B = 256$, image size $I = 32$. single GPU.

Comparison between self-supervised and supervised contrastive loss First, we run different epochs of training using either self-supervised loss or supervised loss. And we report the best validation accuracy during the finetuning stage.

method	training epochs				
	200	400	600	8000	1000
supcon	**71.94**	**74.10**	**71.94**	**73.74**	**74.10**
simclr	66.91	70.14	71.58	70.86	72.30

Table 5: Best validation accuracy (%) during finetuning models trained with different epochs and methods.

It is clear from Fig. 3 that models trained with supervised contrastive loss achieve quicker convergence and better accuracy during the classifier finetuning stage.

Two-stage contrastive learning Following the discussion in Sec. 5.2.3, we adopt a two-stage training schema and then use the same way of finetuning to report the best validation accuracy.

For the simplicity of the table, we use the following notations:

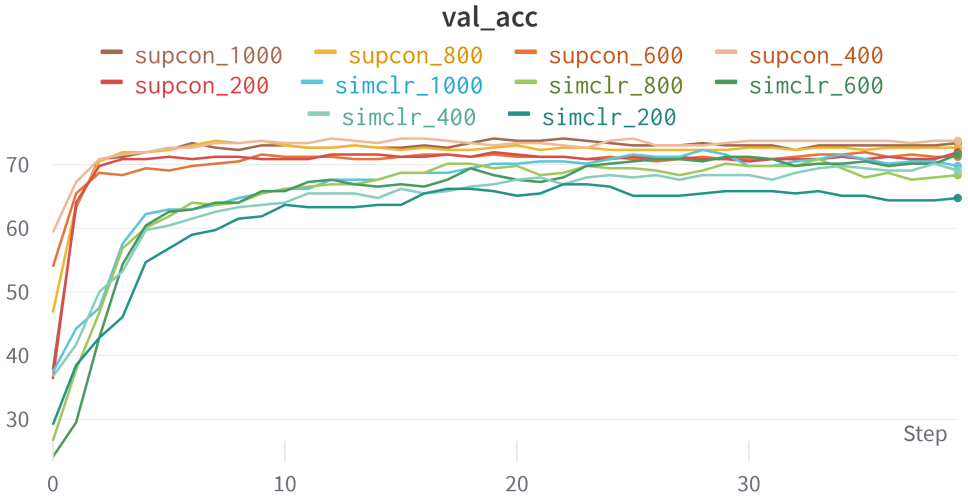


Figure 3: Validation accuracy plot during finetuning models trained with different epochs and methods.

1. n: the number of epochs trained by self-supervised contrastive loss
2. m: the number of epochs trained by supervised contrastive loss
3. $(n : m)$: the model is trained first with n epochs of self-supversied loss and then augmented with m epochs of supervised loss
4. $(m : n)$: the model is trained first with m epochs of supversied loss and then augmented with n epochs of self-supervised loss

We report the best finetuning accuracy on the validation set for each contrastively trained model.

n	m				
	200	400	600	800	1000
0	71.94	**74.10**	**71.94**	**73.74**	**74.10**
50	**72.40**	70.97	68.82	71.32	69.18
100	72.04	70.97	71.33	70.25	72.40
150	68.10	69.89	71.33	68.46	70.61
200	72.04	71.68	69.53	69.53	73.84

Table 6: Best accuracy (%) for $(m : n)$ style two-stage training, with m epochs of supervised contrastive learning first and n extra epochs of self-supvervised contrastive learning.

When first trained with supervised contrastive learning, the auxiliary self-supervised contrastive learning can barely offer any help but instead decrease the accuracy in most situations.

This is expected because supervised contrastive learning provides more useful and accurate training signals than the self-supervised contrastive learning.

Symmetrically,

m	n				
	200	400	600	800	1000
0	66.90	70.14	71.58	70.86	72.30
50	**71.68**	70.61	72.04	71.33	69.18
100	70.97	69.89	**73.48**	**72.40**	71.68
150	69.89	**71.33**	71.33	70.25	**73.48**
200	71.68	70.97	72.76	72.04	71.68

Table 7: Best accuracy (%) for $(n : m)$ style two-stage training, with n epochs of supervised contrastive learning first and m extra epochs of self-supvervised contrastive learning.

But when first trained in a self-supervised way and then with an auxiliary supervised training stage, the latter seems to help improve over the former.

This is also expected in the previous discussion. Self-supervised contrastive learning may be acceptable to minority classes but may not be of much help to majority classes. Then the supervised contrastive learning stage provides more accurate information. Although this method still falls behind the one-stage supervised training method, we still notice some possible improvements. For example, in $(n : m) = (600, 100)$, it achieves 73.48% accuracy, exceeding the supervised training results by a large margin.

We understand that this two-stage training may be inferior to the simple one-stage supervised contrastive learning. But our purpose here is to delve deeper into the difference between these two kinds of methods and how they affect learning on an imbalanced dataset.

6 Conclusion

In this work, we discover the problem of chimpanzee detection and identification. For detection, we perform transfer learning based on public detection models, as well as adjusting the NMS strategy. For identification, a skewed dataset of a small number of samples motivates us to take inspiration from human face recognition, few-shot learning, and imbalanced learning. Mitigating large margin cosine loss designed for human face recognition to our task results in a remarkable performance. Self-supervised contrastive learning and supervised contrastive learning are also compared with both qualitative and quantitative analysis.

In a broader sense, this work suggests that researchers should also pay attention to situations where data collection causes a huge amount of effort and is of low quality or a small amount. Despite the great progress in traditional fields like human face recognition, other real-world applications like chimpanzee recognition systems still pose great challenges to current methods, yet of equal importance as systems for humans. We believe that efforts in this direction will bring more benefits to the whole scientific research field.

References

[1] Github - ultralytics/yolov5: Yolov5 in pytorch. `https://github.com/ultralytics/yolov5`. Accessed 19 January 2023. 2

[2] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019. 2, 3

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 6

[4] Jialue Fan, Wei Xu, Ying Wu, and Yihong Gong. Human tracking using convolutional neural networks. *IEEE transactions on Neural Networks*, 21(10):1610–1623, 2010. 1

[5] Yihui He, Xiangyu Zhang, Marios Savvides, and Kris Kitani. Softer-nms: Rethinking bounding box regression for accurate object detection. *arXiv preprint arXiv:1809.08545*, 2(3):69–80, 2018. 5

[6] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. Learning non-maximum suppression. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4507–4515, 2017. 5

[7] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020. 7

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2

[9] Songtao Liu, Di Huang, and Yunhong Wang. Adaptive nms: Refining pedestrian detection in a crowd. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6459–6468, 2019. 5

[10] Alexander Mathis, Pranav Mamidanna, Kevin M Cury, Taiga Abe, Venkatesh N Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature neuroscience*, 21(9):1281–1289, 2018. 1

[11] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2

[12] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014. 2

[13] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018. 2, 5

[14] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020. 2

[15] Mang Ye, Jianbing Shen, Gaojie Lin, Tao Xiang, Ling Shao, and Steven CH Hoi. Deep learning for person re-identification: A survey and outlook. *IEEE transactions on pattern analysis and machine intelligence*, 44(6):2872–2893, 2021. 1

A Appendix

	train	val
Azibo	227	86
Bambari	5	3
Corrie	2	1
Dorien	38	13
Fraukje	6	5
Frodo	23	8
Kisha	6	3
Kofi	1	0
Lobo	28	10
Lome	26	8
Maja	1	1
Natascha	3	1
Ohini	52	18
Riet	66	38
Sandra	26	18
Swela	151	64
Tai	1	1
(Total)	662	278

Table 8: Details of the identification dataset.