Continual Forgetting for Pre-trained Vision Models
——Supplementary Materials——

Hongbo Zhao[1,3*] Bolin Ni[1,3*] Junsong Fan[2] Yuxi Wang[2]
Yuntao Chen[2] Gaofeng Meng[1,2,3✉] Zhaoxiang Zhang[1,2,3,4✉]

[1]State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences
[2]Centre for Artificial Intelligence and Robotics, Hong Kong Institute of Science & Innovation, Chinese Academy of Sciences
[3] University of Chinese Academy of Sciences [4] Shanghai Artificial Intelligence Laboratory
{zhaohongbo2022, zhaoxiang.zhang}@ia.ac.cn gfmeng@nlpr.ia.ac.cn

Supplementary Material

This document provides the supplementary materials that cannot fit into the main manuscript due to the page limit. Specifically, we first visualize the results on COCO dataset and the group sparsity in Sec. A. Next, we provide more implementation details for reproducibility in Sec. B. Finally, we provide more experiments in Sec. C.

A. Visualization

Visualization of detection results. We provide visualization results on COCO validation set before and after forgetting. Fig. S2 is the result of single-step forgetting and Fig. S3 is the result of continual forgetting. It is observed that GS-LoRA can achieve selective removal without affecting the remaining classes.

Visualization of parameter groups. To show the scalability of GS-LoRA, we evaluate it on Face Transformers with 6 layers, 12 layers and 18 layers in Tab. 7. We visualize the ℓ_2 norm of each LoRA group in these models in Fig. S1. It is observed that GS-LoRA achieves a sparse modification on models of different sizes and shows excellent scalability. Meanwhile, we can find that deeper layers in the Face Transformer contain more class-specific information.

B. Implementation Details

B.1. Face Recognition

Network Architecture. Face Transformer is proposed by Zhong *et al*. [17] who first uses Transformer architecture to solve face recognition tasks. A Face Transformer is a stack of Transformer blocks with a CosFace [14] classifier.

*Equal contribution. ✉ Corresponding author.

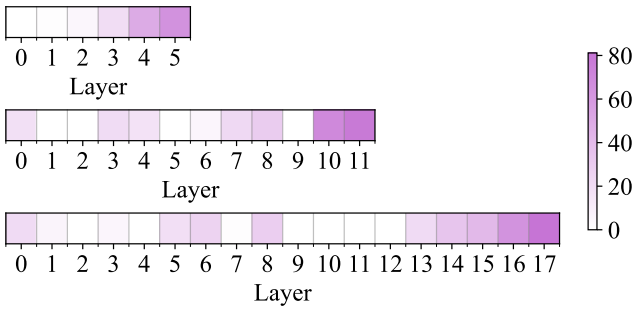

Figure S1. ℓ_2 **norm of each LoRA group in different Face Transformers.** The first row shows a Face Transformer with 6 layers. The second row shows a Face Transformer with 12 layers. The last row shows a Face Transformer with 18 layers. Lighter colors mean smaller ℓ_2 norms which indicate less modification.

Config	Value
optimizer	AdamW
base learning rate	3e-4
learning rate schedule	cosine decay
minimal learning rate	1e-5
weight decay	0.05
momentum	β_1, β_2 =0.9, 0.999
batch size	480
warm-up epochs	10
warm-up learning rate	1e-6
training epochs	1200
dropout rate	0.1

Table S1. Pre-training settings for Face Transformer.

Pre-training. We pre-train a Face Transformer on CASIA-Face100 dataset for 1200 epochs. Implementation details can be found in Tab. S1.

Forgetting. For the forgetting process, we use 1337 as the random seed to generate a forgetting order. Implementa-

Figure S2. Visualization for **single-step forgetting**.

Config	Value
optimizer	AdamW
base learning rate	1e-2
learning rate schedule	cosine decay
minimal learning rate	1e-5
weight decay	0.05
momentum	β_1, β_2 =0.9, 0.999
batch size	48
warm-up epochs	0
training epochs	100
dropout rate	0.1
BND	110
K	20
β	0.15
α_K	0.01
LoRA rank	8
data ratio	0.1

Table S2. Forgetting settings for Face Transformer.

tion details can be found in Tab. S2 for all experimental settings, *which shows the robustness of GS-LoRA*. Here, BND is the bound in Eq. (10), K and α_K is the hyperparameter in Eq. (9), and β is the hyperparameter in Eq. (12). Note that "data ratio" is the ratio of data used for forgetting to data used for pre-training. To achieve fast model editing, the forgetting epoch is set to 100 and 0.1$\times$ pre-training data is used. This is equivalent to fine-tuning the model using all pre-training data with only 10 epochs, which is *less than 1%* compared with 1200 epochs in the pre-training process.

B.2. Object Detection

Network Architecture. We use a Deformable DETR in object detection tasks. Deformable DETR has 3 parts: backbone, encoder and decoder. The backbone is an ImageNet [2] pre-trained ResNet-50 [7]. The encoder and decoder both have 6 Transformer blocks using multi-head deformable attention.

Pre-training. We use the pre-trained model released by

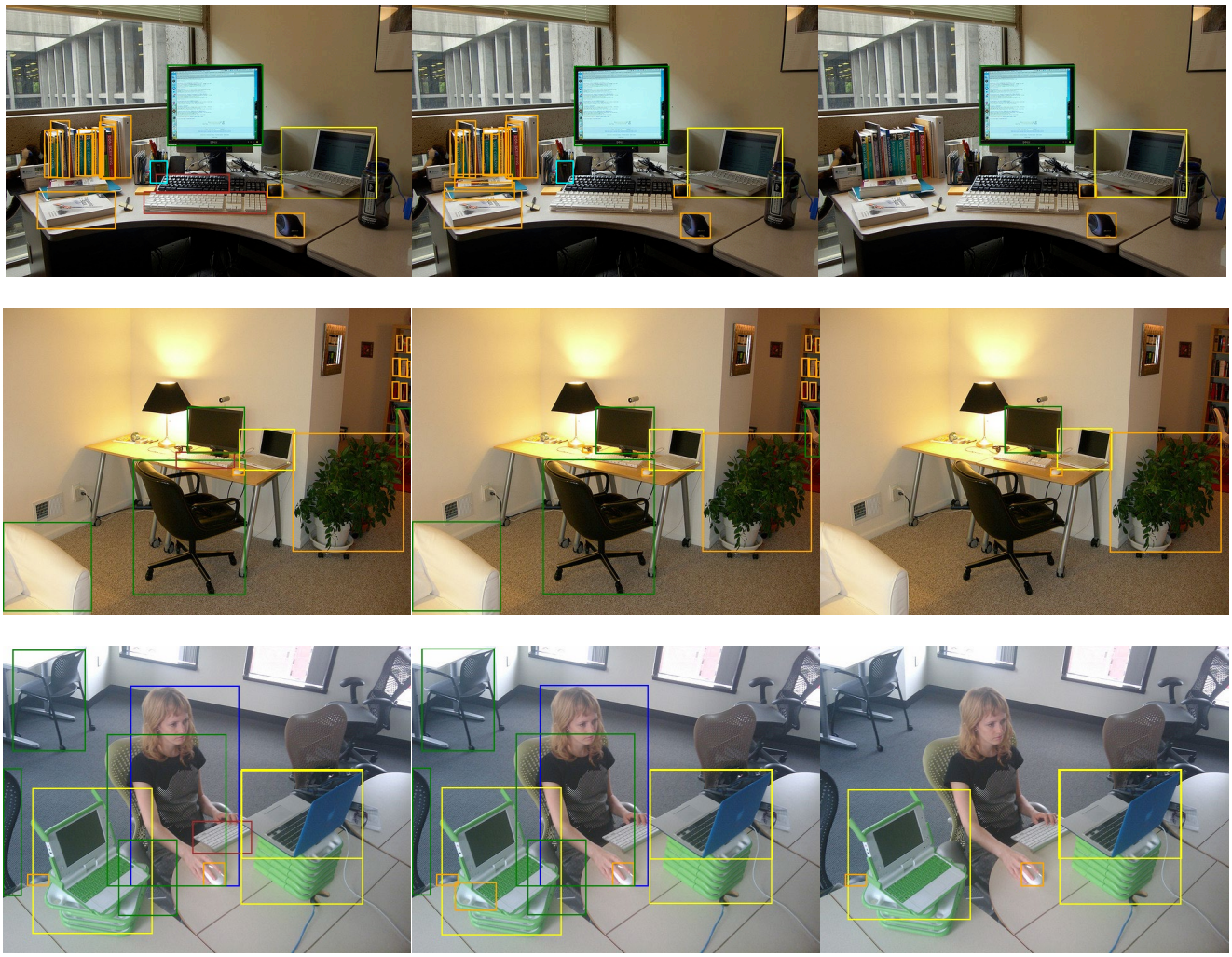

Figure S3. Visualization for **continual forgetting**. The left column shows the results from the pre-trained model. The middle column shows the results when "keyboard" (red bounding boxes in the left column) is erased. The left column shows the results when more objects (*e.g.*, person, book, chair) are erased.

Zhu *et al*. [18], where Deformable DETR is trained on COCO 2017 training set for 50 epochs and reaches 43.8 AP on COCO 2017 validation set.

Forgetting. For the forgetting process, we first generate a random list using seed 123 following Liu *et al*. [11] to determine the forgetting order. Implementation details can be found in Tab. S3 when 40 classes are forgotten. For other experimental settings, hyperparameters are slightly different on β and the learning rate.

B.3. Baselines Implementation Details

B.3.1 Continual Learning Methods

We implement six continual learning methods to realize continual forgetting. Taking EWC as an example, we conduct it for forgetting as follows. First, we give randomly wrong labels to the forget set. Then, we use the remaining set to calculate the weight importance of EWC. Finally, we regard learning on the modified forget dataset as a new task and perform EWC algorithm. Note that we freeze the final FFN layer to ensure backbone forgetting. Additionally, for a fair comparison, we use the remaining set as a replay buffer to enhance the performance, which is denoted as EWC*.

B.3.2 Machine Unlearning Methods

Existing machine unlearning methods can be categorized into exact unlearning and approximate unlearning. Exacting unlearning needs to conduct specific designs in the pre-training process, however, we cannot modify the pre-training process in a continual forgetting setting. Initial studies on approximate unlearning are computationally

Config	Value
optimizer	SGD
momentum	0.9
base learning rate	2e-4
weight decay	1e-4
batch size	16
training epochs	30
dropout rate	0.1
BND	15
β	0.2
α_K	3e-4
gradient clipping	0.1
LoRA rank	8
data ratio	0.1

Table S3. Forgetting settings for Deformable DETR.

heavy, *e.g.*, [4, 6, 12] need to calculate the Hessian matrix. These methods cannot be applied to large-scale problems, and we do not compare with them. We compare our GS-LoRA with state-of-the-art LIRF [15], SCRUB [10] and SCRUB-S (a variant of SCRUB). LIRF [15] also uses a distillation strategy to realize the deposit and withdrawal of knowledge in a model. For a fair comparison, we add an additional replay buffer for LIRF.

SCRUB [10] uses distillation to realize efficient approximate unlearning. The training objective is:

$$\begin{aligned}\min_{w^u} & \frac{\alpha}{N_r} \sum_{x_r \in \mathcal{D}_r} d\left(x_r; w^u\right) \\ & + \frac{\gamma}{N_r} \sum_{\left(x_r, y_r\right) \in \mathcal{D}_r} \ell\left(f\left(x_r; w^u\right), y_r\right) \\ & - \frac{1}{N_f} \sum_{x_f \in \mathcal{D}_f} d\left(x_f; w^u\right),\end{aligned} \tag{S1}$$

where $d\left(x; w^u\right) = D_{\mathrm{KL}}\left(p\left(f\left(x; w^o\right)\right) \| p\left(f\left(x; w^u\right)\right)\right)$ is the KL-divergence between the student (w_u) and teacher (w_o) output distributions for example x, w_o is the weight of a pre-trained model (teacher) and w_u is the student. x_r and x_f are the retained set and forgotten set which contain N_r and N_f samples, respectively. ℓ is the cross-entropy loss and α and γ are hyperparameters.

However, Kurmanji *et al*. [10] find that directly optimizing Eq. (S1) is challenging and utilize a min-max optimization method following GAN [5]. To further improve the performance, we adopt a smoothing optimization method [16] as a variant of SCRUB and name it SCRUB-S.

???? show the results in single-step forgetting and continual forgetting settings. It is observed that LIRF cannot realize effective forgetting under our fast model erasure setting, which is data-inefficient. SCRUB and SCRUB-S can achieve forgetting when a small number of classes need to be deleted, but GS-LoRA achieves better overall performance (H-Mean). When we want to delete a large number of classes, *only GS-LoRA can achieve complete forgetting while maintaining the performance of the rest.* In a continual forgetting setting, SCRUB-S can achieve comparable performance with GS-LoRA, but the accuracy on previously forgotten classes (Acc_o) is a little bit high in SCRUB-S, which is undesirable. In summary, the data efficiency, parameter efficiency and effectiveness of GS-LoRA make it the most applicable in real-world scenarios.

C. More Experiments

In this section, we conduct more experiments to verify the effectiveness and efficiency of GS-LoRA. In Sec. C.1, we perform GS-LoRA in image classification tasks. In Sec. C.2, we conduct ablation studies on β in the loss function. In Sec. C.4, we perform more experiments when the replay buffer is incomplete and compare GS-LoRA with continual learning baselines.

C.1. Experiments on Image Classification

To further demonstrate the universality of our method, we use GS-LoRA to realize continual forgetting on image classification tasks. We choose ImageNet100 [2] dataset and a pre-trained ViT [3] model in S4, where *GS-LoRA still outperforms other baselines significantly*.

C.2. Ablations on β in Loss Function

Our data loss function is **??**, where β controls the level of forgetting. We conduct ablation studies in S5 on face recognition tasks. It is amazing that we find GS-LoRA demonstrates excellent performance *across a wide range of* β, which shows the robustness of our method.

C.3. Different Grouping Strategies

By default, we regard two LoRA modules in a Transformer block as a group (see in **??**). In this section, we explore the effect of using GS-LoRA with different grouping strategies. In the FFN module [13], there are two linear layers, each of which can add a LoRA module. And in a LoRA module [8], there are two low-rank matrices.

We consider three grouping strategies: *"Block"*, *"Module"* and *"Matrix"*. *"Block"* is the default setting. *"Module"* denotes each LoRA **module** is a group, resulting in twice the number of groups compared to the Transformer blocks. *"Matrix"* means each **matrix** in LoRA modules is a group and the number of groups is four times the number of Transformer blocks.

We conduct our experiments on a Face Transformer and all the experiments are performed at the same sparse intensity, *i.e.* α is the same. Tab. S6 shows the results of different grouping strategies. Notably, all grouping strategies yield exceptional performance. It is observed that with a more detailed grouping strategy, the zero-group ratio increases,

Methods	100-20			80-20				60-20				40-20			
	$H\uparrow$	$Acc_r\uparrow$	$Acc_f\downarrow$	$H\uparrow$	$Acc_r\uparrow$	$Acc_f\downarrow$	$Acc_o\downarrow$	$H\uparrow$	$Acc_r\uparrow$	$Acc_f\downarrow$	$Acc_o\downarrow$	$H\uparrow$	$Acc_r\uparrow$	$Acc_f\downarrow$	$Acc_o\downarrow$
Pre-train	-	68.0	60.5	-	64.1	60.9	-	-	69.2	63.3	-	-	76.5	70.7	-
L2*	47.6	**64.9**	22.9	47.6	68.0	24.3	30.0	40.0	70.1	35.3	24.5	52.7	74.7	29.9	23.5
EWC* [9]	54.1	**64.9**	14.2	55.6	64.9	12.2	13.9	53.8	72.7	20.7	7.8	62.0	80.0	20.1	9.1
MAS* [1]	54.0	**64.9**	14.2	57.1	**69.0**	12.2	13.7	54.0	72.7	20.4	7.7	62.2	80.2	19.8	8.8
Retrain	22.1	13.6	**1.0**	33.0	22.7	**0.7**	**0.0**	41.1	31.0	**2.2**	**0.0**	54.7	46.1	**3.4**	**0.0**
GS-LoRA	**60.3**	63.0	2.6	**61.7**	66.9	3.6	0.8	**65.9**	**74.1**	4.0	0.5	**73.8**	**82.7**	4.1	**0.0**

Table S4. **Continual forgetting results for image classification.** Acc_o is the accuracy of old tasks, *i.e.*, the accuracy on all previously forgotten classes in task $\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_{t-1}$. There are 4 tasks in total and 20 classes are forgotten in each task.

β	0.01	0.05	0.1	0.15	0.2	0.25	0.5	0.75	1	5	10	20
$H\uparrow$	60.9	71.5	71.6	**72.2**	71.8	72.0	71.5	72.0	71.8	70.1	70.5	66.3

Table S5. Ablation studies on β.

Grouping Strategy	$Acc_f\downarrow$	$Acc_r\uparrow$	$H\uparrow$	Zero Group Ratio
Block	1.97	71.06	71.43	0.17
Module	0.93	70.58	71.70	0.50
Matrix	1.51	70.36	71.30	0.58

Table S6. **Effect of grouping strategies.** The zero-group ratio goes up when a more detailed grouping strategy is used.

which makes sense because each group has fewer parameters and more flexibility.

C.4. More Experiments with Incomplete Replay

In Sec. 5.3, we consider the situation where we cannot obtain the replay data of some classes. In this section, we conduct more experiments with incomplete replay data on other continual learning baselines. We keep our settings the same as Fig. 5, where 30 classes need to be forgotten. In the remaining 70 classes, some classes cannot be replayed. Tab. S7 shows the results when 5, 10, 15, 20, 25, 30, 35 and 40 classes cannot be replayed. We can find that GS-LoRA mitigates catastrophic forgetting to some extent and achieves the best performance among all listed baselines.

	$Acc_f \downarrow$	$Acc_r \uparrow$	$H \uparrow$	$Acc_r^\dagger \uparrow$
Pre-train	73.67	75.00	-	73.97
L2*	0.04	56.81	64.13	21.06
EWC*	0.04	55.42	63.24	14.56
MAS*	**0.00**	55.20	63.11	16.78
Retrain	**0.00**	14.84	24.70	5.65
LoRA	0.04	**70.07**	**71.81**	49.66
GS-LoRA	0.04	**70.07**	**71.81**	**55.43**

(a) No replay data in 5 of the 70 remaining categories.

	$Acc_f \downarrow$	$Acc_r \uparrow$	$H \uparrow$	$Acc_r^\dagger \uparrow$
Pre-train	73.67	75.11	-	75.99
L2*	0.04	55.61	63.36	31.12
EWC*	**0.00**	51.41	60.56	16.19
MAS*	**0.00**	51.71	60.77	17.63
Retrain	**0.00**	13.48	22.79	4.68
LoRA	**0.00**	67.57	70.49	51.35
GS-LoRA	0.04	**68.49**	**70.97**	**57.37**

(b) No replay data in 10 of the 70 remaining categories.

	$Acc_f \downarrow$	$Acc_r \uparrow$	$H \uparrow$	$Acc_r^\dagger \uparrow$
Pre-train	73.67	74.83	-	76.44
L2*	0.04	51.65	60.71	27.32
EWC*	**0.00**	45.86	56.53	8.31
MAS*	0.04	47.07	57.43	12.07
Retrain	**0.00**	9.12	16.23	0.13
LoRA	**0.00**	66.64	69.98	54.12
GS-LoRA	**0.00**	**66.85**	**70.09**	**58.14**

(c) No replay data in 15 of the 70 remaining categories.

	$Acc_f \downarrow$	$Acc_r \uparrow$	$H \uparrow$	$Acc_r^\dagger \uparrow$
Pre-train	73.67	74.79	-	76.66
L2*	0.00	48.67	58.62	24.32
EWC*	0.00	41.32	52.95	6.55
MAS*	0.00	41.56	53.14	7.64
Retrain	0.00	8.14	14.66	0.10
LoRA	0.00	63.83	68.40	49.27
GS-LoRA	0.00	**64.45**	**68.75**	**54.37**

(d) No replay data in 20 of the 70 remaining categories.

	$Acc_f \downarrow$	$Acc_r \uparrow$	$H \uparrow$	$Acc_r^\dagger \uparrow$
Pre-train	73.67	74.81	-	75.88
L2*	**0.00**	44.53	55.50	22.14
EWC*	**0.00**	38.28	50.38	7.73
MAS*	**0.00**	37.81	49.97	8.24
Retrain	**0.00**	8.03	14.48	0.17
LoRA	**0.00**	64.40	68.72	54.41
GS-LoRA	0.04	**66.12**	**69.68**	**59.87**

(e) No replay data in 25 of the 70 remaining categories.

	$Acc_f \downarrow$	$Acc_r \uparrow$	$H \uparrow$	$Acc_r^\dagger \uparrow$
Pre-train	73.67	74.89	-	74.70
L2*	0.00	40.86	52.56	18.77
EWC*	0.00	33.56	46.11	5.07
MAS*	0.00	35.82	48.20	8.21
Retrain	0.00	7.13	13.00	0.50
LoRA	0.00	58.10	64.96	41.61
GS-LoRA	0.00	**59.52**	**65.84**	**46.25**

(f) No replay data in 30 of the 70 remaining categories.

	$Acc_f \downarrow$	$Acc_r \uparrow$	$H \uparrow$	$Acc_r^\dagger \uparrow$
Pre-train	73.67	74.82	-	75.02
L2*	0.00	37.32	49.55	18.23
EWC*	0.00	29.66	42.30	4.24
MAS*	0.00	30.04	42.68	5.66
Retrain	0.00	5.67	10.53	0.09
LoRA	0.00	52.27	61.15	34.16
GS-LoRA	0.00	**56.47**	**63.93**	**43.46**

(g) No replay data in 35 of the 70 remaining categories.

	$Acc_f \downarrow$	$Acc_r \uparrow$	$H \uparrow$	$Acc_r^\dagger \uparrow$
Pre-train	73.67	75.04	-	74.70
L2*	0.00	33.29	45.86	16.06
EWC*	0.00	25.08	37.41	4.15
MAS*	0.00	25.22	37.57	4.67
Retrain	0.00	6.02	11.13	0.30
LoRA	0.00	58.24	65.05	46.48
GS-LoRA	0.00	**62.84**	**67.83**	**54.29**

(h) No replay data in 40 of the 70 remaining categories.

Table S7. **Experiment results with incomplete replay.** Thirty classes are forgotten in all experiments. In the remaining 70 classes, only some classes can be replayed. Each subtable shows the results with a different number of replay classes. Pre-train denotes the results before forgetting. L2*, MAS* and EWC* denote the original methods with a rehearsal buffer. LoRA denotes using LoRA to fine-tune FFN modules in Transformer blocks without group sparse. Our method (GS-LoRA) is highlighted in color. We specifically evaluate the accuracy of the classes without replay samples and report it as $Acc_r^\dagger$.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pages 139–154, 2018. 5

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 4

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[4] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9304–9312, 2020. 4

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 4

[6] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019. 4

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[8] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 4

[9] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 5

[10] Meghdad Kurmanji, Peter Triantafillou, and Eleni Triantafillou. Towards unbounded machine unlearning. *Advances in neural information processing systems*, 2023. 4

[11] Yaoyao Liu, Bernt Schiele, Andrea Vedaldi, and Christian Rupprecht. Continual detection transformer for incremental object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23799–23808, 2023. 3

[12] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. *Advances in Neural Information Processing Systems*, 34:18075–18086, 2021. 4

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4

[14] 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. 1

[15] Jingwen Ye, Yifang Fu, Jie Song, Xingyi Yang, Songhua Liu, Xin Jin, Mingli Song, and Xinchao Wang. Learning with recoverable forgetting. In *ECCV*, 2022. 4

[16] Jiawei Zhang, Peijun Xiao, Ruoyu Sun, and Zhiquan Luo. A single-loop smoothed gradient descent-ascent algorithm for nonconvex-concave min-max problems. *Advances in neural information processing systems*, 33:7377–7389, 2020. 4

[17] Yaoyao Zhong and Weihong Deng. Face transformer for recognition. *arXiv preprint arXiv:2103.14803*, 2021. 1

[18] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3