Self-Supervised Models are Continual Learners (Supplementary Material)

Enrico Fini*[1,2] Victor G. Turrisi da Costa*[1] Xavier Alameda-Pineda[2]
Elisa Ricci[1,3] Karteek Alahari[2] Julien Mairal[2]

[1] University of Trento [2] Inria[†] [3] Fondazione Bruno Kessler

Appendix

A. PyTorch-like pseudo-code

We provide a PyTorch-like pseudo-code of our method. As you can see, CaSSLe is simple to implement and does not add much complexity to the base SSL method. In this snippet, the losses are made symmetric by summing the two contributions. In some cases, the two losses are averaged instead. In CaSSLe, we symmetrize in the same way as the base SSL method we are considering.

Algorithm 1 PyTorch-like pseudo-code for CaSSLe.

```
# aug: stochastic image augmentation
# f: backbone and projector
# frozen_f: frozen backbone and projector
# g: CaSSLe's predictor
# loss_fn: any SSL loss in Tab. 1 (main paper)

# PyTorchLightning handles loading and optimization
def training_step(x):

    # correlated views
    x1, x2 = aug(x), aug(x)

    # forward backbone and projector
    z1, z2 = f(x1), f(x2)

    # optionally forward predictor...

    # compute SSL loss (symmetric)
    ssl_loss = loss_fn(z1, z2) \\
             + loss_fn(z2, z1)

    # forward frozen backbone and projector
    z1_bar, z2_bar = frozen_f(x1), frozen_f(x2)

    # compute distillation loss (symmetric)
    distill_loss = loss_fn(g(z1), z1_bar) \\
                 + loss_fn(g(z2), z2_bar)

    # no hyperparameter for loss weighting
    return ssl_loss + distill_loss
```

*Enrico Fini and Victor G. Turrisi da Costa contributed equally.
†Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK, 38000 Grenoble, France.

B. Derivation of distillation losses

In this section, we derive distillation losses from the SSL losses in Tab. 1 of the main paper, starting from the definition of our distillation loss:

$$\mathcal{L}_D(\boldsymbol{z}, \bar{\boldsymbol{z}}) = \mathcal{L}_{SSL}(g(\boldsymbol{z}), \bar{\boldsymbol{z}}), \tag{1}$$

where z and $\bar{z}$ are the representations of the current and frozen encoder, and g is CaSSLe's predictor network implemented as a two layer MLP with 2048 hidden neurons and ReLU activation.

Contrastive based. Our distillation loss based on contrastive learning is implemented as follows:

$$\mathcal{L}(z_i, \bar{z}_i) = -\log \frac{\exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \bar{\boldsymbol{z}}_i\right)/\tau\right)}{\sum_{\boldsymbol{z}_j \in \bar{\eta}(i)} \exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_j\right)/\tau\right)}, \tag{2}$$

where $\bar{\eta}(i)$ is the set of negatives for the sample with index i in the batch. Note that the negatives are drawn both from the predicted and frozen features.

MSE based. This distillation loss is simply the MSE between the predicted features and the frozen features:

$$\mathcal{L}(z, \bar{z}) = -||g(\boldsymbol{z}) - \bar{\boldsymbol{z}}||_2^2. \tag{3}$$

It can be implemented with the cosine similarity as stated in the main manuscript.

Cross-entropy based. The cross-entropy loss, when used for distillation in an unsupervised setting, makes sure that the current encoder is able to assign samples to the frozen centroids (or prototypes) consistently with the frozen encoder:

$$\mathcal{L}(z, \bar{z}) = -\sum_d \bar{\boldsymbol{a}}_d \log \frac{\exp\left(\operatorname{sim}\left(g(\boldsymbol{z}), \boldsymbol{c}_d^{t-1}\right)/\tau\right)}{\sum_k \exp\left(\operatorname{sim}\left(g(\boldsymbol{z}), \boldsymbol{c}_k^{t-1}\right)/\tau\right)} \tag{4}$$

where:

$$\bar{\boldsymbol{a}} = \frac{\exp\left(\mathrm{sim}\left(\bar{\boldsymbol{z}}, \boldsymbol{c}_d^{t-1}\right)/\tau\right)}{\sum_k \exp\left(\mathrm{sim}\left(\bar{\boldsymbol{z}}, \boldsymbol{c}_k^{t-1}\right)/\tau\right)}, \tag{5}$$

and the set of frozen prototypes is denoted as follows: $\mathbf{C}^{t-1} = \left\{\mathbf{c}_1^{t-1}, \ldots, \mathbf{c}_K^{t-1}\right\}$.

Cross-correlation based. We consider Barlow Twins' [27] implementation of this objective. For VICReg [3] we only consider the invariance term. As a distillation loss, the cross-correlation matrix is computed with the predicted and frozen features:

$$\mathcal{L}(z, \bar{z}) = \sum_u \left(1 - \bar{\mathcal{C}}_{uv}\right)^2 + \lambda \sum_u \sum_{v \neq u} \bar{\mathcal{C}}_{uv}^2, \tag{6}$$

where:

$$\bar{\mathcal{C}}_{uv} = \frac{\sum_i g(\boldsymbol{z}_{i,u}) \bar{\boldsymbol{z}}_{i,v}}{\sqrt{\sum_i g\left(\boldsymbol{z}_{i,u}\right)^2} \cdot \sqrt{\sum_i \left(\bar{\boldsymbol{z}}_{i,v}\right)^2}}. \tag{7}$$

C. Further discussion and implementation details of the baselines

Selection. When evaluating our framework, we try to compare with as many existing related methods as possible. However, given that SSL models are computationally intensive, it was not possible to run all baselines and methods in all the CL settings we considered. As mentioned in the main manuscript, we choose eight baselines (seven related methods + fine-tuning) belonging to three CL macro-categories, and test them on CIFAR100 (class-incremental) in combination with three SSL methods. The selection was based on the ease of adaptation to CSSL and the similarity to our framework.

The most similar to CaSSLe are data-focused regularization methods. Among them, a large majority leverage knowledge distillation using the outputs of a classifier learned with supervision *e.g.* [7, 12, 20], while a few works employ feature distillation [10, 16] which is viable even without supervision. [17] is also related to CaSSLe, but it focuses on memory efficiency which is less interesting in our setting. Also, [17] explicitly uses the classifier after feature adaptation, hence it is unclear how to adapt it for CSSL, especially since in SSL positives are generated using image augmentations, which are not applicable to a memory bank of features. On the contrary, augmentations can be used in replay methods, among which we select the most common (ER [24]) and one of the most recent (DER [4]). Regarding prior-focused regularization methods, we choose EWC [19] over others (SI [28], MAS [2], *etc.*) as it is considered the most influential and it works best with task boundaries. We also consider two CSSL baselines: LUMP [22] and Lin *et al.* [21]. Finally, we do not consider methods based on VAEs [1, 23], since they have been shown to yield poor performance in the large and medium scale. For instance, as found by [11], a VAE trained offline on CIFAR10 reaches an accuracy of 57.2%, which is lower than any method (except VICReg) trained continually on CIFAR100 with CaSSLe.

Table A. Linear evaluation top-1 accuracy on ImageNet100 (5 tasks, class- and data-incremental).

Method	Strategy	ImageNet100	
		Class-inc.	Data-inc.
Supervised	Fine-tuning	61.6	74.3
Contrastive	CaSSLe	**69.6**	**76.9**

Implementation. For EWC, we use the SSL loss instead of the supervised loss to estimate importance weights. For POD and Less-Forget, we only re-implement the feature distillation without considering the parts of their methods that explicitly use the classifier. For DER, we replace the logits of the classifier with the projected features in the buffer. We re-implement all these baselines by adapting them from the official implementation (POD), or from the Mammoth framework provided with [4] (DER, ER, EWC), or from the paper (Less-Forget). We also compare with two concurrent works that propose approaches for CSSL (LUMP [22], Lin *et al.* [21]). LUMP uses k-NN evaluation, therefore we adapt the code provided by the authors to run in our code base. For Lin *et al.*, we compare directly with their published results, since they use the same evaluation protocol. We perform hyperparameter tuning for all baselines, searching over 5 values for the distillation loss weights of POD and Less-Forget, 3 values for the weight of the regularization in EWC and 3 replay batch sizes for replay methods. The size of the replay buffer is 500 samples for all replay based methods.

D. Additional results

Continual supervised contrastive with CaSSLe. After the popularization of contrastive learning [8, 14] for unsupervised learning of representations, [18] proposed a supervised version of the contrastive loss. Here, we show that CaSSLe is easily extendable to support supervised contrastive learning. The implementation is basically the same as for our vanilla contrastive-based distillation loss. In Tab. A, we show the improvement that CaSSLe brings with respect to fine-tuning, which is sizeable in the class-incremental setting. We also report the same comparison

Table B. Linear evaluation top-1 accuracy on DomainNet (6 tasks, domain-incremental setting) w/ and w/o CaSSLe. The sequence of tasks is Real→Quickdraw→Painting→Sketch→Infograph→Clipart. "Aw." stands for task-aware, "Ag," for task-agnostic.

Method	**Strategy**	**Real**		**Quickdraw**		**Painting**		**Sketch**		**Infograph**		**Clipart**		**Avg.**	
		Aw.	Ag.	Aw.	Ag.	Aw.	Ag.	Aw.	Ag.	Aw.	Ag.	Aw.	Ag.	Aw.	Ag.
Barlow Twins	Finetuning	56.3	50.9	54.1	45.8	42.7	35.9	49.0	41.9	22.0	17.4	59.0	52.5	50.3	43.7
	CaSSLe	**62.7**	**57.1**	**59.1**	**50.6**	**49.2**	**42.1**	**53.8**	**47.7**	**25.5**	**20.6**	**61.9**	**55.6**	**55.5**	**48.9**
	Offline	67.1	63.0	60.3	53.9	52.4	46.3	51.9	46.9	25.9	21.0	58.8	52.6	57.2	51.8
SwAV	Finetuning	57.7	52.3	53.2	43.5	43.0	35.9	46.1	39.0	21.6	16.5	53.4	46.6	49.6	42.5
	CaSSLe	**62.8**	**57.8**	**59.5**	**50.2**	**47.5**	**41.2**	**49.5**	**42.5**	**22.5**	**17.9**	**56.5**	**49.6**	**54.3**	**47.5**
	Offline	64.1	59.5	60.6	53.6	47.6	42.9	47.7	42.1	23.3	18.9	53.6	47.3	54.6	49.1
BYOL	Finetuning	58.7	53.2	51.7	41.6	44.0	37.4	49.6	43.9	23.5	19.0	58.6	53.5	50.6	43.8
	CaSSLe	**63.7**	**60.5**	**59.3**	**50.9**	**48.6**	**44.1**	**50.4**	**45.2**	**24.1**	**19.4**	**59.0**	**54.4**	**55.1**	**49.7**
	Offline	67.2	64.0	60.2	53.3	51.5	47.3	50.4	46.2	24.5	20.8	57.0	51.5	56.6	51.9
VICReg	Finetuning	54.7	49.6	53.0	44.9	42.1	34.7	49.0	41.9	21.1	16.4	58.5	52.6	49.3	42.8
	CaSSLe	**59.0**	**53.2**	**56.4**	**47.8**	**46.0**	**38.9**	**52.3**	**45.6**	**23.9**	**18.5**	**60.9**	**55.3**	**52.9**	**46.1**
	Offline	66.4	62.7	59.2	53.5	52.4	47.2	53.2	48.1	25.3	20.7	58.3	53.2	56.7	51.9
SimCLR	Finetuning	52.5	47.6	48.2	38.1	37.5	31.7	42.8	35.7	18.8	14.4	50.9	46.8	45.1	38.4
	CaSSLe	**58.4**	**43.4**	**54.2**	**44.7**	**43.9**	**37.7**	**47.6**	**41.9**	**22.0**	**17.8**	**54.9**	**50.5**	**50.0**	**44.2**
	Offline	62.1	59.5	58.3	52.9	46.1	42.5	45.6	41.3	22.1	18.8	51.0	45.9	52.6	48.6
MoCoV2+	Finetuning	50.9	45.5	45.8	37.5	36.0	29.3	39.5	32.1	17.9	13.5	50.3	**44.5**	43.2	36.7
	CaSSLe	**56.0**	**50.3**	**48.7**	**40.0**	**40.4**	**33.6**	**42.0**	**35.0**	**19.9**	**15.2**	**51.7**	**44.5**	**46.7**	**38.8**
	Offline	65.2	61.3	57.9	51.3	48.7	43.1	44.7	39.1	23.4	19.0	51.3	44.8	53.7	48.4
Supervised Contrastive	Finetuning	57.7	52.6	55.3	45.5	44.9	38.0	51.7	45.0	22.6	18.3	64.0	60.0	52.1	45.4
	CaSSLe	**63.4**	**58.8**	**59.7**	**51.3**	**50.1**	**44.7**	**55.9**	**50.3**	**26.9**	**22.4**	**65.0**	**61.3**	**56.7**	**50.9**
	Offline	67.4	65.3	65.8	63.0	53.6	50.9	56.0	53.1	28.0	25.7	62.8	59.6	60.0	57.4
Supervised	Finetuning	63.0	58.2	56.9	47.6	49.1	44.0	55.7	50.3	27.7	23.3	68.6	63.5	55.9	49.8
	Offline	74.7	73.2	68.5	67.8	62.0	59.3	65.7	63.7	33.7	34.5	72.3	69.3	66.4	65.0

on DomainNet in Tab. B, showing interesting results in both task-aware and task-incremental evaluation.

Task-agnostic evaluation and domain-wise accuracy on DomainNet. In the main manuscript, we showed that CaSSLe significantly improved performance in the domain-incremental setting using task-aware evaluation. Here, "task-aware" refers to the fact that linear evaluation is performed on each domain separately, *i.e*. a different linear classifier is learned for each domain. However, it might also be interesting to check the performance of the model when the domain is unknown at test time. For this reason, we report the performance of our model when evaluated in a task-agnostic fashion. In addition, we also show the accuracy on each task (*i.e*. domain). All this information is presented in Tab. B. CaSSLe **always** outperforms fine-tuning with both evaluation protocols. The accuracy of CaSSLe on "Clipart" is also higher than offline. This is probably due to a combination of factors: (i) Clipart is the last task, therefore it probably benefits in forward transfer and (ii) a similar effect to the one found in [26], where dividing data in subgroups tends to enable the learning of better representations. Also, we notice that task-agnostic accuracy is lower than the task-aware counterpart. This is expected and

Table C. k-NN evaluation on ImageNet100 (5 tasks, class-incremental) performed on backbone and projected features.

Method	**Strategy**	**k-NN accuracy (↑)**	
		Backbone (f_b)	**Projector (f_p)**
Barlow Twins	Fine-tuning	59.1	34.4
	CaSSLe	**63.4**	**53.2**
SwAV	Fine-tuning	**60.0**	53.9
	CaSSLe	59.7	**61.3**
BYOL	Fine-tuning	57.1	33.0
	CaSSLe	**61.2**	**60.8**
VICReg	Fine-tuning	56.7	35.3
	CaSSLe	**59.5**	**43.4**
MoCoV2+	Fine-tuning	54.5	39.0
	CaSSLe	**61.5**	**53.1**
SimCLR	Fine-tuning	54.8	40.1
	CaSSLe	**61.7**	**53.2**

means that the class conditional distributions are not perfectly aligned in different domains. As in the main paper, the colors are related to the type of SSL loss.

Additional results with k-NN evaluation. For completeness, in this supplementary material, we also show that CaSSLe yields superior performance when evaluated with a k-NN classifier instead of linear evaluation. We use

Table D. Linear evaluation top-1 accuracy on CIFAR100 (10 tasks, class-incremental).

Method	Strategy	A (↑)
SimCLR	Fine-tuning	39.3
	CaSSLe	**52.7**
Barlow Twins	Fine-tuning	49.9
	CaSSLe	**53.7**

Table E. Linear evaluation top-1 accuracy on ImageNet100 (5 tasks, class- and data-incremental) with ResNet50 [15].

Method	Strategy	A (↑)	
		Class-inc.	Data-inc.
SimCLR	Fine-tuning	70.7	75.6
	CaSSLe	**74.0**	**77.2**
Barlow Twins	Fine-tuning	71.2	75.8
	CaSSLe	**74.8**	**78.1**

weighted k-NN with l2-normalization (cosine similarity) and temperature scaling as in [6]. Since since k-NN is much faster than linear evaluation we could also assess the quality of the projected representations, instead of just using the backbone. The results can be inspected in Tab. C. Three interesting phenomena arise: (i) CaSSLe always improves with respect to fine-tuning, (ii) the features of the backbone f_b are usually better than the features of the projector f_p and (iii) CaSSLe causes information retention in the projector, which significantly increases the performance of the projected features. An exception is represented by SwAV [5], that seems to behave differently to other methods. First, the accuracy of the projected features in SwAV is much higher than other methods. This might be due to the fact that it uses prototypes, which bring the representations 1 layer away from the loss, making them less specialized in the SSL task. Second, it seems that CaSSLe only improves the projected features when coupled with SwAV. However, this is probably an artifact of the evaluation procedure, as the l2-normalization probably causes loss of information. Indeed, although the overall performance is lower, SwAV + CaSSLe outperforms SwAV + fine-tuning (58.7% vs 56.9%) if the euclidean distance is used in place of the cosine similarity for the backbone features. We leave a deeper investigation of this phenomenon for future work.

Different number of tasks. The analysis of CSSL settings that we show in the main manuscript is limited to the 5 task scenario. However, it is interesting to run the same benchmarks with a longer task sequence. Nonetheless, one should also remember that SSL methods are data hungry, hence the less data is available per task, the higher the instability of the SSL models. In Tab. D, we present additional results with 10 tasks on CIFAR100 (class-incremental).

Table F. Combinations of SSL methods and distillation losses on CIFAR100 (class-incremental, 2 tasks).

Distillation Loss	SimCLR	Barlow Twins	BYOL
InfoNCE	**61.8**	64.5	64.8
Cross-correlation	60.1	**67.2**	65.8
MSE	61.3	64.6	**66.7**

Barlow Twins seems to hold up surprisingly well, finishing up at roughly 50% accuracy, while SimCLR suffers in the low data regime. Nonetheless, CaSSLe outperforms fine-tuning with Barlow Twins, and to a very large extent with SimCLR.

Deeper architectures. The experiments we propose in the main manuscript feature a ResNet18 network. This is a common choice in CL. However, in SSL, it is more common to use ResNet50. For this reason, in Tab. E we show that the same behavior observed with smaller networks is also obtained with deeper architectures. More specifically, CaSSLe outperforms fine-tuning in both class- and data-incremental settings by large margins.

The role of the predictor. In the main manuscript, we provided an intuitive explanation of the role of the predictor network that maps the current feature space to the frozen feature space. This intuition is corroborated by extensive experimentation and ablation studies. However, one more thing that is worth mentioning is that the success of the predictor network might also be related to the findings in SimSiam [9], BYOL [13] and DirectPred [25]. Moreover, we perform additional ablations on the design of CaSSLe's predictor for SimCLR on CIFAR100 (5 tasks): adding BatchNorm after the hidden layer does not make any difference in terms of performance, and removing the non-linearity only causes a 0.3% drop in accuracy.

Combinations of SSL methods and distillation losses. For computational reasons, it was not feasible to perform experiments combing all SSL methods with all possible distillation losses. However, in Tab. F we provide a subset of the possible combinations to validate our strategy that uses the same SSL loss for distillation.

References

[1] Alessandro Achille, Tom Eccles, Loic Matthey, Christopher P Burgess, Nick Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. *NeurIPS*, 2018. 2

[2] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, 2018. 2

[3] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021. 2

[4] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *NeurIPS*, 2020. 2

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020. 4

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *ICCV*, 2021. 4

[7] Francisco M Castro, Manuel J Marin-Jimenez, Nicolas Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, 2018. 2

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 2

[9] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *ICCV*, 2021. 4

[10] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, 2020. 2

[11] William Falcon, Ananya Harsh Jha, Teddy Koker, and Kyunghyun Cho. Aavae: Augmentation-augmented variational autoencoders. *arXiv preprint arXiv:2107.12329*, 2021. 2

[12] Enrico Fini, Stèphane Lathuilière, Enver Sangineto, Moin Nabi, and Elisa Ricci. Online continual learning under extreme memory constraints. In *ECCV*, 2020. 2

[13] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *NeurIPS*, 2020. 4

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 4

[16] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, pages 831–839, 2019. 2

[17] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *European Conference on Computer Vision*, pages 699–715. Springer, 2020. 2

[18] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *NeurIPS*, 2020. 2

[19] 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. *Proc. of the national academy of sciences*, 2017. 2

[20] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 2017. 2

[21] Zhiwei Lin, Yongtao Wang, and Hongxiang Lin. Continual contrastive self-supervised learning for image classification. *arXiv preprint arXiv:2107.01776*, 2021. 2

[22] Divyam Madaan, Jaehong Yoon, Yuanchun Li, Yunxin Liu, and Sung Ju Hwang. Rethinking the representational continuity: Towards unsupervised continual learning. *arXiv preprint arXiv:2110.06976*, 2021. 2

[23] Dushyant Rao, Francesco Visin, Andrei A Rusu, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Continual unsupervised representation learning. *NeurIPS*, 2019. 2

[24] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 1995. 2

[25] Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning*, pages 10268–10278. PMLR, 2021. 4

[26] Yonglong Tian, Olivier J Henaff, and Aaron van den Oord. Divide and contrast: Self-supervised learning from uncurated data. *arXiv preprint arXiv:2105.08054*, 2021. 3

[27] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *ICML*, 2021. 2

[28] Friedeman Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, 2017. 2