HOW WELL DOES SELF-SUPERVISED PRE-TRAINING PERFORM WITH STREAMING DATA?

Anonymous authors
Paper under double-blind review

ABSTRACT

Prior works on self-supervised pre-training focus on the joint training scenario, in which massive unlabeled data are assumed to be given as input all at once and only then a learner is trained. Such a problem setting, unfortunately, is often impractical if not infeasible at all, since many real-world tasks rely on sequential learning, e.g., data are decentralized or collected in a streaming fashion. In this paper, we conduct a thorough and the first dedicated investigation on self-supervised pre-training with streaming data, aiming to shed light on the model behavior under this overlooked setup. Specifically, we pre-train over 500 models on four categories of pre-training streaming data from ImageNet and DomainNet and evaluate them on three types of downstream tasks and 12 different downstream datasets. Our studies show that, somehow beyond our expectation, with simple data replay or parameter regularization, sequential self-supervised pre-training turns out to be an efficient alternative for joint pre-training, as the performances of the former are mostly on par with those of the latter. Moreover, catastrophic forgetting, a common issue in sequential supervised learning, is much alleviated in sequential self-supervised learning (SSL). Our findings, therefore, suggest that, in practice, for the SSL paradigm, the cumbersome joint training can be largely replaced by sequential learning, which in turn enables a much broader spectrum of potential application scenarios. Code and models will be released.

1 INTRODUCTION

Recent advances in self-supervised learning (SSL) (He et al., 2020; Grill et al., 2020; Caron et al., 2020; Jure et al., 2021) demonstrate competitive or even better transfer learning performance on various downstream tasks, compared with supervised pre-training. Although waiving the cost of human labeling, SSL usually requires massive unlabeled data to learn a powerful representation model and benefits from significantly large-scale pre-training data, e.g., He et al. (2020) adopted billion-scale data to pre-train better SSL models. The common pre-training practice follows the **joint training** (JT) setup, where massive unlabeled data are collected together before model training. In reality, however, it is usually difficult to access a large amount of collective unlabeled data at once. Instead, real-world data are usually accessed in a streaming fashion, e.g., data are generated and collected sequentially chunk by chunk (Delange et al., 2021), or even decentralized and stored in different servers (Lange et al., 2020); such a learning setup is known as **sequential training** (ST). Despite much research effort and promising results achieved by JT, it inevitably suffers from heavy data storage, prolonged training time, and finds itself incompetent when training data volume expands over time. For ST, on the other hand, a learner can be sequentially trained with disjoint data chunks, making it much more efficient than JT.

How to effectively and efficiently pre-train a representation model under the ST setup has been an open problem. Despite the high efficiency, some continual learning research works (Goodfellow et al., 2013; Kirkpatrick et al., 2017; Rebuffi et al., 2017) have shown that ST with supervised models tends to suffer from catastrophic forgetting (McCloskey & Cohen, 1989), having a significant performance degradation on the historical data chunks. Unlike the case of continual learning tasks, in pre-training tasks, one expects the model to well generalize to downstream tasks rather than focusing only on the seen tasks (Chen et al., 2020a). Nevertheless, how well sequential self-supervised models perform on downstream tasks remains unclear.

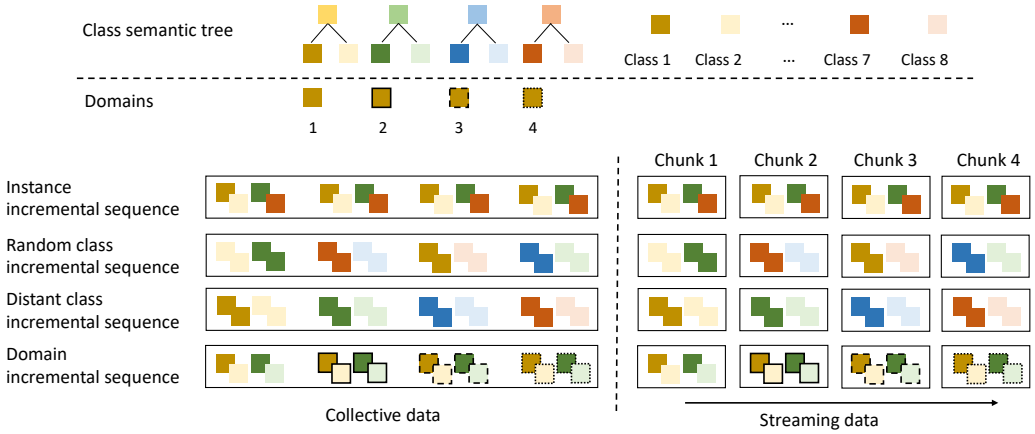


Figure 1: Illustration of four types of streaming data with various distribution shifts and the corresponding collective data. Different colors represent different classes, where similar colors refer to similar semantics in the class semantic tree. Different border types refer to different domain styles.

To fill the research gap, we provide a thorough empirical study on the transfer learning behavior of SSL models pre-trained with streaming data. In the pre-training stage, to mimic real-world data collection scenarios with different degrees of distribution shift, as shown in Figure 1, we consider four types of streaming data chunks, including (1) the instance incremental sequence with negligible data distribution shift, (2) the random class incremental sequence with moderate data distribution shift, (3) the distant class incremental sequence with severe data distribution shift, and (4) the domain incremental sequence with severe domain distribution shift. The first three types of streaming data are based on 1,000-class ImageNet (Russakovsky et al., 2015) and the last domain incremental sequence is composed of diverse domains in DomainNet (Peng et al., 2019).

As for the evaluation, we consider three downstream tasks following Ericsson et al. (2021), including few-shot evaluation and linear evaluation (also named many-shot classification) on 12 image classification datasets (Kornblith et al., 2019b), and the Pascal VOC (Everingham et al., 2010) detection task. Through extensive experiments with more than 500 pre-trained models, we thoroughly investigate key roles in sequential SSL, including streaming data, downstream tasks and datasets, continual learning methods, SSL methods, and the method efficiency in terms of time and storage. We also provide a thorough investigation on the knowledge forgetting behavior of sequential SSL models and sequential SL models.

To the best of our knowledge, we are among the first to explore the sequential self-supervised pre-training setting and the first to provide a thorough empirical study on self-supervised pre-training with streaming data. We summarize the takeaways as well as our contributions as follows:

- Sequential SSL models exhibit almost the same transfer learning performance as joint SSL models on streaming data with moderate distribution shift. As for streaming data with severe distribution shift, i.e., the distant class sequence and the domain sequence, there exist evident performance gaps between sequential SSL models and joint SSL models. Such performance gaps, however, can be mitigated effectively and efficiently with unsupervised parameter regularization (Aljundi et al., 2018) and simple data replay.

- Based on the above finding, the common joint training paradigm may be unnecessary for SSL to obtain a good representation model. Instead, sequential SSL is performance-competitive but more time-efficient and storage-saving and is well worth considering as a practical paradigm for self-supervised pre-training with streaming data.

- In sequential training, compared with supervised learning (SL) models, SSL models consistently show smaller performance gaps between ST and JT. Moreover, sequential SSL models are less prone to catastrophic forgetting than sequential SL models. We believe such properties of SSL will inspire potential applications of SSL in continual learning tasks.

2 Related Work

Self-supervised learning (SSL). SSL learns useful features by solving various pretext tasks using supervisions generated from unlabeled training data, e.g., predicting rotations (Gidaris et al., 2018), solving jigsaw puzzles (Noroozi & Favaro, 2016), predicting colorization (Larsson et al., 2016), predicting cluster assignments (Caron et al., 2018), and solving instance discrimination (Wu et al., 2018; Chen et al., 2020a; He et al., 2020; Grill et al., 2020). To achieve better performance, recent studies of SSL have leveraged especially large datasets for pre-training, including YFCC 100M (Thomee et al., 2016) and Instagram 1B (Mahajan et al., 2018). However, in realistic scenarios, access to massive data is often in a streaming fashion and how to perform SSL with streaming data has not been studied before, motivating our work.

Continual learning. Existing studies of continual learning (CL) (Delange et al., 2021) mainly focus on supervised tasks and can be summarized into three categories, including regularization, replay, and parameter-isolation. In regularization-based CL, knowledge preserving is achieved by regularizing the parameter posterior of the new task not to deviate drastically from the prior (Aljundi et al., 2018; Kirkpatrick et al., 2017; Zenke et al., 2017). Replay-based CL methods overcome forgetting by saving samples of previous tasks in a replay buffer (Rebuffi et al., 2017; Rolnick et al., 2019; Lopez-Paz & Ranzato, 2017; Wang et al., 2021) and using them to regularize the learning of new tasks. Last, isolation-based CL methods leverage different parameters for learning each task to preserve the learned knowledge (Serra et al., 2018; Mallya & Lazebnik, 2018). Although works (Rao et al., 2019; Aljundi et al., 2019) explore continual learning for some specific unsupervised tasks, few have studied the transfer learning performance of sequential self-supervised models.

3 Problem Setting

In pre-training tasks, we train representation models on large-scale datasets, such as ImageNet (Russakovsky et al., 2015), and evaluate the transferability of learned representations on various downstream tasks (Chen et al., 2020a). In our empirical study, we adopt the prevailing MoCo-v2 (Chen et al., 2020c) method to pre-train SSL models with diverse streaming data.

Types of streaming data. As for pre-training data, we consider four types of streaming data to mimic practical data collection scenarios. As shown in Figure 1, each type of streaming data consists of sequential and disjoint data chunks, while collective data cover all available data. (1) In the instance incremental sequence, streaming data chunks are almost independent identically distributed (IID), which simulates the scenario where IID data are continually collected under the same condition. In this case, there is negligible distribution shift among sequential data chunks. (2) In the random class incremental sequence, data in disjoint chunks belong to different classes, which mimics the scenario where data are collected by random keyword search on the Internet (Parisi et al., 2019). There exists moderate distribution shift within this data sequence. (3) The distant class incremental sequence is similar to the random class incremental sequence except that the distant class incremental sequence has larger semantic gaps between sequential data chunks, i.e., images from different data chunks are semantically dissimilar. This data sequence has severe distribution shift between sequential data chunks. It mimics the scenario where data are crawled from websites with different subjects. (4) In the domain incremental sequence, data chunks are collected from different domains with severe domain distribution shift. A typical example is that large-scale autonomous driving data in Han et al. (2021) are collected in different domains, such as different weather conditions and cities, but share similar classes across domains. The first three types of streaming data are designed ImageNet (Russakovsky et al., 2015), while the domain incremental sequence consists of five domains in DomainNet (Peng et al., 2019). See Appendix A for a detailed description.

Model pre-training. With these streaming data, we study both sequential training (ST) and joint training (JT) for model pre-training. As illustrated in Figure 1, in sequential training, a model is sequentially trained with streaming data chunks, while in joint training, a model is repeatedly retrained with collective data, i.e., all seen data chunks. Moreover, we compare SSL with supervised learning (SL) and mainly study the following pre-trained models: sequentially trained SSL models (SSL-ST), jointly trained SSL models (SSL-JT), sequentially trained SL models (SL-ST), and jointly trained SL models (SL-JT). See Appendix B for details of the pre-training stage.

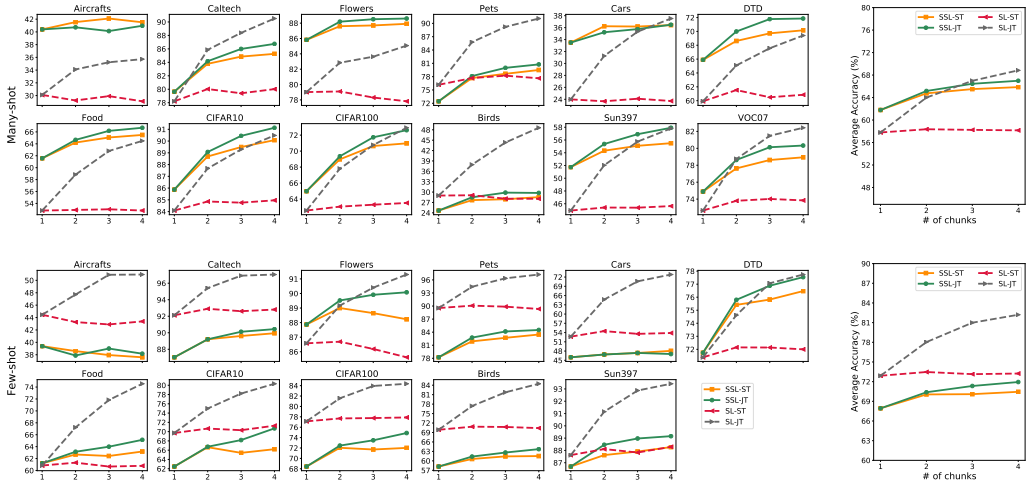


Figure 2: Linear and few-shot evaluation results of **random class incremental sequence**. On the left are the results of each dataset. On the right are averaged results across all left datasets.

Transfer to downstream tasks. We evaluate the transfer learning performance of pre-trained models using three typical downstream tasks: many-shot classification, few-shot classification, and object detection. Following Chen et al. (2020a), we consider 12 classification datasets for downstream evaluation. Specifically, we conduct many-shot classification on all the above 12 datasets but conduct few-shot classification on 11 datasets except VOC2007, following Ericsson et al. (2021). In both types of classification tasks, representations are fixed for evaluation. In addition, we evaluate pre-trained models on the PASCAL VOC detection dataset following He et al. (2020). See Appendix C for more details of the downstream evaluation.

4 Dissection of Sequential Self-Supervised Pre-Training

In this section, we provide a comprehensive empirical study on SSL models pre-trained with streaming data. Specifically, we study several key factors affecting the downstream transfer learning performance of sequential SSL models, including types of streaming data in Section 4.1, downstream tasks and datasets in Section 4.2, continual learning methods in Section 4.3, and the adopted SSL methods in Section 4.4. In addition, we analyze the time efficiency and the storage efficiency of models trained with streaming data in Section 4.5.

We pre-train representation models on four types of streaming data and evaluate pre-trained models on 12 downstream datasets with three downstream evaluation tasks. Note that models pre-trained with ImageNet-based streaming data are evaluated on all three downstream tasks. Models trained with the domain incremental sequence are only evaluated with few-shot classification, considering that the size of each data chunk in DomainNet is only $1/5$ that of each chunk in ImageNet. We report downstream evaluation results of the random class incremental sequence, the distant class incremental sequence, the domain incremental sequence in Figure 2, Figure 3, and Figure 5, respectively. Results of the instance incremental sequence are illustrated in Appendix F since the performance of SSL models on the instance incremental sequence is similar to that on the random class incremental sequence without considering the different cases for SL models. We also evaluate three types of ImageNet-based streaming data on object detection and illustrate results in Figure 4.

4.1 How does transfer learning performance vary with streaming data?

We first consider streaming data with various degrees of distribution shift, i.e., streaming data with negligible distribution shift such as the instance incremental sequence, streaming data with moderate distribution shift such as the random class incremental sequence, and streaming data with severe distribution shift such as the distant class incremental sequence and the domain incremental sequence. As shown in Figures 2-5, on all types of streaming data, the performance of sequential SSL models

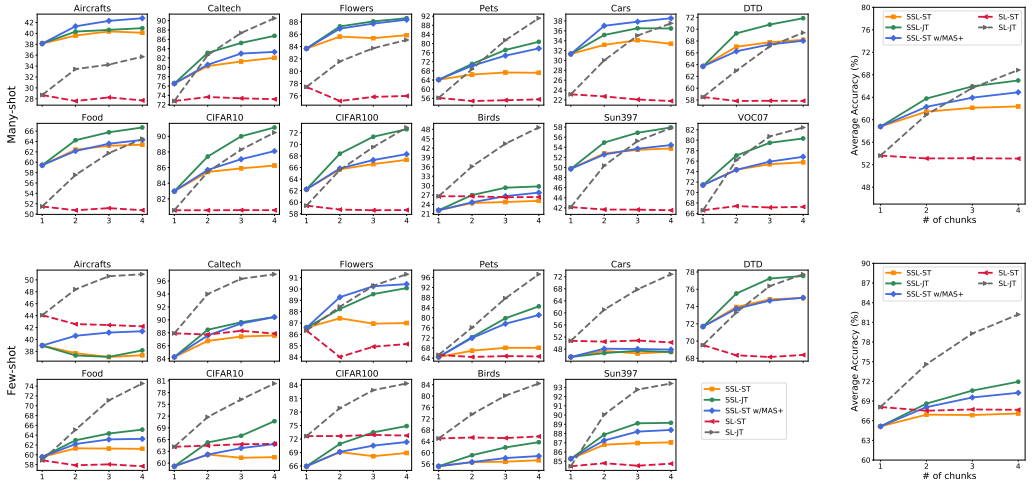
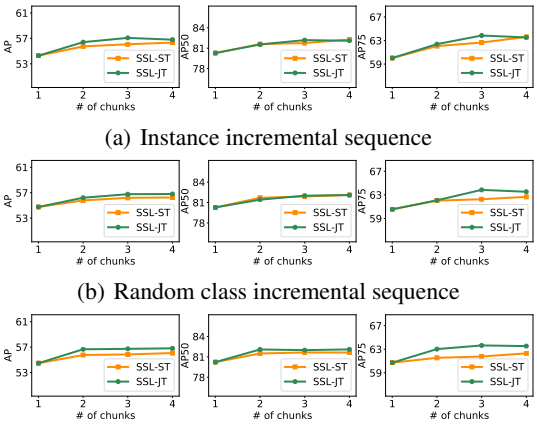


Figure 3: Linear and few-shot evaluation results of **distant class incremental sequence**. on the left are the results of each dataset. On the right are averaged results across all left datasets.

generally increases with more streaming chunks, while sequential SL models do not benefit from increasing data chunks. As for the performance on each type of streaming data, sequential SSL models surprisingly perform comparably to joint SSL models on streaming data with negligible and moderate distribution shift. On streaming data with severe distribution shift, the performance of sequential SSL models is evidently inferior to that of joint SSL models. The detection evaluation results in Figure 4 further support this observation. In contrast, on all types of streaming data, sequential SL models perform obviously worse than joint SL models. The above observations denote that, unlike traditional continual learning tasks (Delange et al., 2021), although faced with visible performance gaps between ST models and JT models on streaming data with severe distribution shift, sequential SSL is still performance-promising in pre-training tasks with streaming data.

4.2 Are results consistent across downstream tasks or datasets?

In pre-training tasks, we pay attention to the generalization ability of the learned models to new data or tasks, rather than the performance on the training dataset. As shown in Figures 2-5, sequential SSL is performance-promising across the three downstream tasks, supported by the average results across all datasets. We also have another consistent observation that SL models show more substantial advantages than SSL models in few-shot transfer learning tasks, while joint SSL models achieve comparable performance to SL models in linear evaluation. Two previous studies (Tian et al., 2020; Ericsson et al., 2021) demonstrated similar observations to ours. The underlying reason may be that supervised features are discriminative while self-supervised features are more generalized. In addition, on some downstream datasets, we have seemingly abnormal observations that ST models may outperform JT models and the model performance may drop with the increase of chunk number. These phenomena are due to the so-called "negative transfer" (Wang et al., 2019), which is also discussed in other model pre-training studies (Newell & Deng, 2020; Gururangan et al., 2020). That is, pre-training with more data chunks generally improves the model ability, but does not necessarily benefit a specific downstream dataset if the added training data are



(a) Instance incremental sequence

(b) Random class incremental sequence

(c) Distant class incremental sequence

Figure 4: Object detection evaluation results of three types of ImageNet-based streaming data.

irrelevant to the downstream task. Compared with sequential SSL, it is observed that sequential SL suffers more severe "negative transfer" on the performance of downstream tasks.

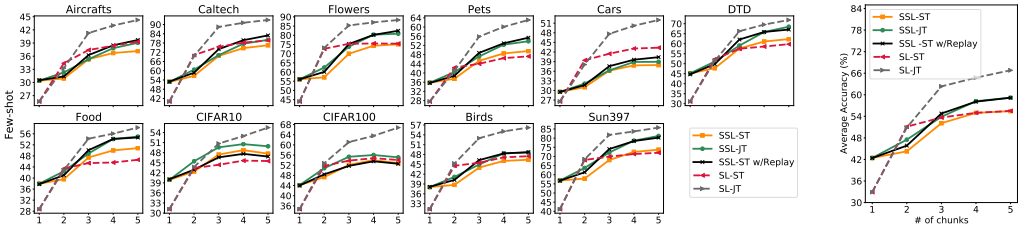


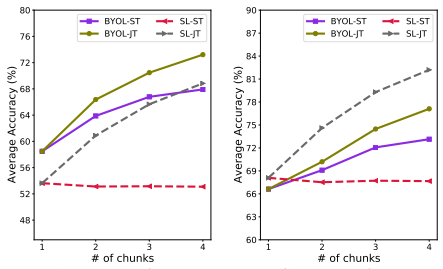
Figure 5: Few-shot evaluation results of **domain incremental sequence**. on the left are the results of each dataset. On the right are averaged results across all left datasets.

4.3 Do continual learning methods help sequential SSL?

As observed in Section 4.1, there still exist obvious performance gaps between sequential SSL models and joint SSL models, on streaming data with severe distribution shift such as the distant class incremental sequence and the domain incremental sequence. To this end, we study whether continual learning methods can help mitigate such gaps. Specifically, we investigate two classic methods in continual learning, i.e., data replay and MAS (Aljundi et al., 2018), which are effective to defy knowledge forgetting in supervised classification tasks (Delange et al., 2021). When using data replay, we randomly reserve 10% data from each seen data chunk and add them to the current data chunk for model pre-training. We also consider the combination of MAS and data replay, which is referred to as MAS+ in the experiments. We denote SSL models trained with data replay as SSL-ST w/Replay, SSL models trained with MAS as SSL-ST w/MAS, and SSL models trained with both methods as SSL-ST w/MAS+. We report downstream evaluation results of the distant class incremental sequence in Figure 3 and results of the domain incremental sequence in Figure 5. As shown in Figure 5, data replay can totally eliminate the performance gaps between sequential SSL models and joint SSL models on the domain incremental sequence. Results in Figure 3 also validate the effectiveness of both continual learning methods in improving the transfer learning performance of sequential SSL models when faced with streaming data with severe distribution shift. In short, we find methods devised for supervised continual tasks are especially promising to make sequential SSL models perform comparably to joint SSL models on challenging streaming data. See Appendix D for implementations of MAS and data replay in sequential SSL.

4.4 How about SSL methods other than MoCo?

For simplicity, we choose MoCo-v2 (Chen et al., 2020c) in experiments and illustrate sequential SSL is performance-promising. To verify whether it also holds for other SSL methods, we train BYOL (Grill et al., 2020) models on the challenging distant class incremental sequence, both sequentially and jointly. Summarized results of BYOL are shown in Figure 6. Similar to MoCo-v2, there still exist visible performance gaps between sequential SSL models and corresponding joint SSL models. In contrast, SSL models exhibit much smaller performance gaps than SL models, which further validates the potential of sequential SSL in pre-training tasks. See detailed results in Appendix G.



(a) Many-shot (b) Few-shot

Figure 6: The averaged downstream evaluation results of BYOL models pre-trained with the distant class incremental sequence.

4.5 Analysis of method efficiency

We then discuss the time and memory consumption of different training methods of SSL, including sequential training (SSL-ST), ST with data replay (SSL-ST w/Replay), ST with MAS (SSL-ST

w/MAS), ST with MAS and data replay (SSL-ST w/MAS+), and joint training (SSL-JT). As shown in Table 1, JT is very time-consuming especially when the data amount is large, while ST is able to save a large amount of time under sequential training scenarios. To be specific, ST is about 2x faster than JT when there are 2 chunks of data, and is about 4x faster when the number of chunks is 4. Moreover, when we use MAS and data replay to improve the performance of ST, the time consumption of SSL increases a little but is still significantly faster than JT. As for storage consumption, we can observe a similar phenomenon as shown in Table 1. In summary, sequential SSL is much more time-efficient and storage-saving than JT, especially when the data amount is large or grows quickly. Such a result indicates that sequential SSL is a more favorable choice for real-world pre-training applications, where data come in sequentially and grow daily.

Table 1: Resource efficiency of considered SSL pre-training methods. We take the distant class incremental sequence as an example and report the training time (h) and required storage (GB) of the model pre-trained with each data chunk. Note that all the following statistics are recorded under the same hardware environment. The lower value means better efficiency.

Time (Storage) / Chunk	**2**	**3**	**4**
SSL-ST	**16.5 (35)**	**16.5 (35)**	**16.6 (35)**
SSL-ST W/Replay	17.0 (35)	18.5 (42)	20.0 (46)
SSL-ST w/MAS	18.2 (35)	18.1 (35)	18.1 (35)
SSL-ST w/MAS+	22.4 (39)	24.4 (42)	26.4 (46)
SSL-JT	31.1 (70)	46.5 (105)	66.6 (140)

Table 2: The comparison of pre-training methods in terms of the transfer performance gap between ST and JT models. We report the averaged accuracy gaps of linear evaluation across 12 downstream datasets. The lower, the better.

Accuracy gap (%) / Chunk	**2**	**3**	**4**
SL-ST (Instance)	2.26	3.27	4.83
SSL-ST (Instance)	**0.41**	**1.02**	**1.04**
SL-ST (Random)	5.63	8.73	10.68
SSL-ST (Random)	**0.42**	**0.94**	**1.13**
SL-ST (Distant)	7.77	12.50	15.75
SSL-ST (Distant)	2.34	3.81	4.62
SSL-ST w/MAS (Distant)	1.82	2.73	3.17
SSL-ST w/MAS+ (Distant)	**1.47**	**2.01**	**2.10**

Summary. We exhibit the averaged accuracy gaps between ST models and the corresponding JT models under linear evaluation in Table 2, for both SSL and supervised learning (SL). On streaming data with negligible distribution shift, SL exhibits evident accuracy gaps while SSL has negligible gaps. On streaming data with moderate distribution shift, SL exhibits larger accuracy gaps while SSL still keeps the negligible gaps. On streaming data with severe distribution shift, SL shows much larger accuracy gaps, while SSL shows mild accuracy gaps. But such accuracy gaps of SSL can be effectively mitigated with simple continual learning methods. To sum up, on streaming pre-training data, sequential SSL is much more promising than sequential SL to substitute joint training, because SSL exhibits significantly smaller performance gaps between ST models and JT models than SL. The above difference between SL and SSL models motivates us to further investigate the forgetting property in Section 5.

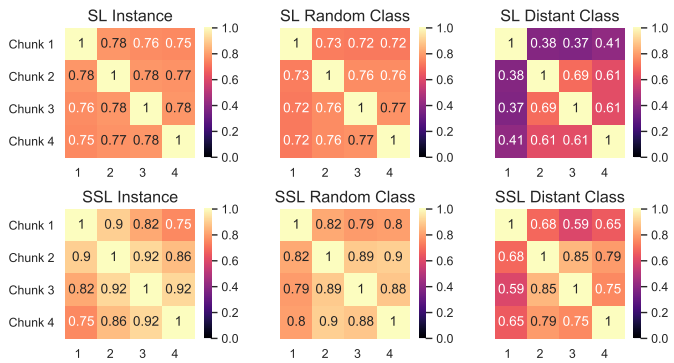
5 Self-Supervised Models Forget Less than Supervised Models

5.1 Backward and Forward Transfer Analysis of Sequential Learning

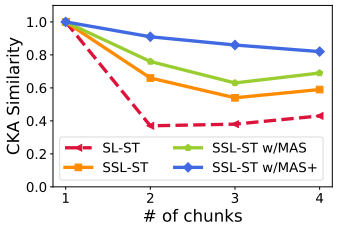
Given the representation, we follow Wu et al. (2018) and evaluate its quality by performing k-Nearest Neighbor (kNN) classification. To provide more insight, we adopt the backward transfer BWT $= \frac{1}{T-1}\sum_{i=2}^{T}\frac{1}{i}\sum_{j=1}^{i} A^{i}_{\mathcal{Y}_j} - A^{j}_{\mathcal{Y}_j}$ and forward transfer FWT $= \frac{1}{T-1}\sum_{i=2}^{T} A^{i}_{\mathcal{Y}_i} - \tilde{A}_{\mathcal{Y}_j}$ metrics used in Yan et al. (2021) where T is the sequence length, $A^{i}_{\mathcal{Y}_j}$ refers to the accuracy on the chunk j using model learned at step i where the label space includes all observed classes up to chunk j, and $\tilde{A}_{\mathcal{Y}_j}$ means the accuracy with the model learned from scratch. The results are shown in Table 3, and we can obtain the following observations about forgetting: *(1) Learning method:* SSL itself is less prone to catastrophic forgetting than SL, especially that SSL achieves positive backward transfer on instance incremental sequence. It illustrates that SSL is more suitable for streaming data; *(2) Types of streaming data:* The model suffers progressively severe forgetting when the distribution shift increases for both SSL and SL

Table 3: Backward and forward transfer analysis of sequential learning.

Data	Method	BWT(%)		FWT(%)	
		Top-1	Top-5	Top-1	Top-5
Instance	SL	-9.45	-5.46	**8.64**	2.81
	SSL	**3.61**	**3.60**	7.55	**8.63**
Random	SL	-20.63	-7.03	-0.34	0.01
	SSL	**-5.17**	**-1.36**	**11.05**	**4.52**
Distant	SL	-40.43	-28.66	4.90	0.47
	SSL	**-13.24**	**-11.06**	**11.01**	**3.66**



(a) CKA scores between sequentially trained models.



(b) CKA similarity scores between the sequentially trained models and the corresponding jointly trained models at the step of each data chunk on the distant class incremental sequence.

Figure 7: CKA similarity analysis of sequentially trained models. Given images in the first data chunk, figure (a) shows the similarity of features between different sequential models on three types of ImageNet-based streaming data. With the same images, figure (b) shows the similarity of features between ST models and the corresponding JT models w.r.t. each data chunk on the distant class incremental sequence. The higher CKA similarity value, the more similar.

cases. *(3) Example forgetting:* It is observed that forgetting is less severe in top-5 classification than top-1 classification, which indicates that the knowledge is not fully forgotten.

5.2 Features memorization analysis of sequential learning

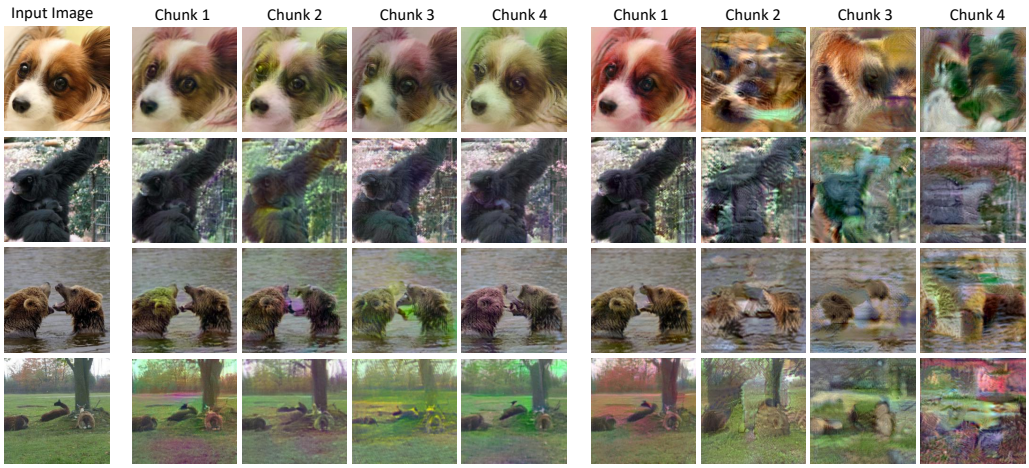
To further understand sequential SSL, we analyze features in the sequential training process from three perspectives: forgetting of initial features, the similarity between ST models and JT models, and reconstructions of original images from the features.

How do features forget in sequential training? We first study how learned features forget in sequential training via the Centered Kernel Alignment (CKA) (Kornblith et al., 2019a). CKA is usually used to measure the similarity between two representations of the same given samples. See Appendix E for details of the CKA similarity. Specifically, we randomly sample 50,000 images from the first data chunk on each type of streaming data. We use these samples and the sequentially trained models for CKA similarity analysis. We report the CKA similarity values on three types of ImageNet-based streaming data in Figure 7(a). Each value in Figure 7(a) is obtained by first extracting features of samples with two different models and then computing the CKA feature similarity value between the two features. On all streaming data, we have three consistent observations about the CKA similarity between sequential models: (1) SSL models all exhibit higher features similarity to the initial model, compared with SL models. (2) In general, SSL models show higher features similarity between two sequential models in sequential training, compared with SL models. (3) Features similarity between two sequential models decrease on streaming data with more severe distribution shift, for both SSL and SSL. These observations suggest that features of SSL models forget less information and evolve more slowly than those of SL models in sequential training.

How are ST models similar to JT models? We then evaluate CKA similarity, for each data chunk, between features from the sequentially trained model and features from the corresponding jointly trained model. The same 50,000 samples are used for CKA features similarity analysis. For example, as shown in Figure 7(b), at the step of the second data chunk, we compute the CKA similarity value between features of the model jointly trained with the first two data chunks and features from the model sequentially trained after the second data chunk. The corresponding CKA similarity value is 0.4, which indicates for SL, the difference between the ST model and the JT model is very large. In contrast, SSL has a higher similarity of 0.7 between the ST model and the JT model. Particularly, with MAS and data replay, the CKA similarity increases to about 0.9, which means the model trained by sequential SSL extracts nearly the same features as the jointly trained model does. Since the analysis is on the streaming data with severe distribution shift, this further reinforces our hy-

pothesis that, with the help of suitable continual learning methods, sequential SSL pre-training is promising to replace joint training on streaming data with various distribution shift.

Feature reconstructions of sequential models. Similar to Zhao et al. (2020), in Figure 8, we visualize feature reconstructions of both sequential SL models and sequential SSL models using deep image prior (DIP) (Ulyanov et al., 2018). To be specific, we choose four images in the first data chunk of the challenging distant class incremental sequence and visualize features of four sequentially learned models for both SSL and SL, respectively. As shown in Figure 8, in sequential training, features of SSL models can always perfectly reconstruct the main information in original images. In contrast, features of SL models lose more detailed information with the increase of data chunks, which indicates SSL is much better at countering the knowledge forgetting in sequential training. Recalling the evolving CKA similarity shown in Figure 7(a), the perfect reconstruction results of sequential SSL models do not mean SSL models stop learning in sequential training, but it indicates that SSL does well in learning new knowledge while keeping previous knowledge.



Raw image | Images reconstructed from SSL models | Images reconstructed from SL models

Figure 8: Feature reconstructions of both SSL models and SL models in sequential pre-training.

6 Discussions

In this paper, we have conducted the first thorough empirical evaluation to investigate how well self-supervised learning (SSL) performs under sequential training scenarios. Our results show the two main findings as follows:

Joint training is not necessary for SSL, while sequential training with suitable strategies is a good alternative. In the scenarios where distribution shift within streaming data is moderate (e.g., instance incremental sequence and random class incremental sequence), it is more favorable to directly conduct sequential SSL training that is far more efficient with negligible loss of transfer learning performance. On the other hand, if distribution shift between streaming data chunks is severe, sequential SSL training with suitable continual learning methods is well worth considering.

Sequential self-supervised pre-training shows a better capability of overcoming catastrophic forgetting than sequential supervised pre-training. One possible reason is that features learned by contrastive SSL have been shown to be uniformly distributed over the feature space (Wang & Isola, 2020), which means the learned representations evolve more slowly during sequential training, as demonstrated by Section 5.2. In addition, features learned by the self-supervised task of instance discrimination are able to keep more information than the features learned by supervised pre-training (Zhao et al., 2020), which weakens the effect of forgetting during sequential training.

As for future directions, we first call for more attention to sequential self-supervised learning for understanding its underlying theories of knowledge forgetting and devising better approaches. Also, we recommend considering sequential self-supervised training as a more efficient representation learning paradigm for real-world applications.

REFERENCES

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *European Conference on Computer Vision*, 2018.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Computer Vision and Pattern Recognition*, 2019.

Thomas Berg, Jiongxin Liu, Seung Woo Lee, Michelle L Alexander, David W Jacobs, and Peter N Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In *Computer Vision and Pattern Recognition*, 2014.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems*, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, 2020a.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In *Advances in Neural Information Processing Systems*, 2020b.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020c.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Computer Vision and Pattern Recognition*, 2014.

Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Linus Ericsson, Henry Gouk, and Timothy M Hospedales. How well do self-supervised models transfer? In *Computer Vision and Pattern Recognition*, 2021.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL visual object classes (VOC) challenge. *International Journal of Computer Vision*, 2010.

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *Computer Vision and Pattern Recognition Workshop*, 2004.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

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. In *Advances in Neural Information Processing Systems*, 2020.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020.

Jianhua Han, Xiwen Liang, Hang Xu, Kai Chen, Lanqing Hong, Chaoqiang Ye, Wei Zhang, Zhenguo Li, Chunjing Xu, and Xiaodan Liang. SODA10M: Towards large-scale object detection benchmark for autonomous driving. *arXiv preprint arXiv:2108.12178*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Computer Vision and Pattern Recognition*, 2020.

Zbontar Jure, Jing Li, Misra Ishan, LeCun Yann, and Deny Stephane. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, 2021.

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*, 2017.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, 2019a.

Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Computer Vision and Pattern Recognition*, 2019b.

Jonathan Krause, Jia Deng, Michael Stark, and Li Fei-Fei. Collecting a large-scale dataset of fine-grained cars. In *Workshop on Fine-Grained Visual Categorization*, 2013.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

Matthias De Lange, Xu Jia, Sarah Parisot, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Unsupervised model personalization while preserving privacy and scalability: An open problem. In *Computer Vision and Pattern Recognition*, 2020.

Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European Conference on Computer Vision*, 2016.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, 2017.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *European Conference on Computer Vision*, 2018.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Computer Vision and Pattern Recognition*, 2018.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*. Elsevier, 1989.

George A Miller. *WordNet: An Electronic Lexical Database*. MIT press, 1998.

Alejandro Newell and Jia Deng. How useful is self-supervised pretraining for visual tasks? In *Computer Vision and Pattern Recognition*, 2020.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, 2016.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *Computer Vision and Pattern Recognition*, 2012.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *International Conference on Computer Vision*, 2019.

Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. In *Advances in Neural Information Processing Systems*, 2019.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Computer Vision and Pattern Recognition*, 2017.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, 2019.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, 2018.

Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. YFCC100M: The new data in multimedia research. *Communications of the ACM*, 2016.

Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, 2020.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Computer Vision and Pattern Recognition*, 2018.

Liyuan Wang, Kuo Yang, Chongxuan Li, Lanqing Hong, Zhenguo Li, and Jun Zhu. Ordisco: Effective and efficient usage of incremental unlabeled data for semi-supervised continual learning. In *Computer Vision and Pattern Recognition*, 2021.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, 2020.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. In *Computer Vision and Pattern Recognition*, 2019.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Computer Vision and Pattern Recognition*, 2018.

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer Vision and Pattern Recognition*, 2010.

Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Computer Vision and Pattern Recognition*, 2021.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, 2014.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of Machine Learning Research*, 2017.

Nanxuan Zhao, Zhirong Wu, Rynson WH Lau, and Stephen Lin. What makes instance discrimination good for transfer learning? In *International Conference on Learning Representations*, 2020.

APPENDIX

A TYPES OF STREAMING DATA

We consider four kinds of streaming data for the study, i.e., the instance incremental sequence, the random class incremental sequence, the distant class incremental sequence, and the domain incremental sequence. To exclude the effect of the number of images, we make sure that data chunks in the same sequence have almost the same data amount. Here we provide more details about these data sequences.

Instance incremental sequence. For the instance incremental sequence, we split the ImageNet (Russakovsky et al., 2015) training data that consists of 1.28 million images with 1,000 classes into four even chunks. We ensure that each data chunk includes the same 1,000 classes with the same number of images for each class, which means these data chunks are independent and identically distributed (IID).

Random class incremental sequence. For the random class incremental sequence, we randomly split the 1,000 classes of ImageNet into four parts where each part has 250 classes. Since each class of ImageNet has around 1,000 images, we can directly obtain four data chunks with almost the same amount of images.

Distant class incremental sequence. To explore the data sequence with severe distribution shift among data chunks, we consider the distant class incremental sequence. Following (Yosinski et al., 2014), rather than randomly splitting the 1,000 classes, we leverage the WordNet Tree (Miller, 1998) to obtain four even data chunks sharing the minimal semantic overlapping. We first build a 1000*1000 adjacent matrix among the 1,000 classes by setting the value of similar classes as 1 and the value of dissimilar classes as 0. To be specific, we take classes sharing the common parent node beneath the ninth depth in the WordNet Tree as similar classes and vice versa. Using the semantic similarity described in the adjacent matrix, we then split the 1,000 classes into independent connected components as shown in Figure 9. Finally, we merge these imbalanced components into four almost even data chunks as shown in Table 4.

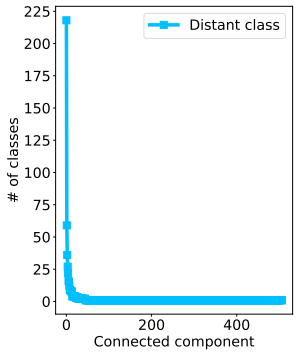


Figure 9: The number of classes for each connected component from the adjacent matrix of 1,000 classes in ImageNet.

Domain incremental sequence. As for the domain incremental sequence, we consider a multi-domain dataset called DomainNet (Peng et al., 2019). In our work, we adopt a domain incremental data sequence made of four distant domains including 'sketch', 'real', 'painting' 'quickdraw', and 'clipart'. There exist severe domain distribution shift among data in these five domains. Specifically, data in the domain 'quickdraw' mostly contain only lines without visual textures. As a result, images from 'quickdraw' are less informative and more visually distinct, compared with images from those four domains, as shown in Figure 10. For each domain, we randomly select 48,129 images as a data chunk, except for 'quickdraw' where we select 47,687 images.

Table 4: The statistics for the four data chunks in the distant class incremental sequence.

	1	2	3	4
Classes	250	250	251	249
Images	318,459	321,488	321,533	319,687

B DETAILS OF PRE-TRAINING

MoCo-v2. For the illustration purpose, we adopt a prevailing self-supervised learning (SSL) method, MoCo-v2 (Chen et al., 2020c), to investigate the performance of SSL with streaming data. MoCo-v2 uses a Siamese network consisting of two encoders. These two encoders are designed for

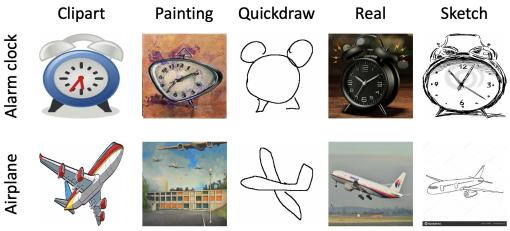


Figure 10: Example images in the five domains of DomainNet.

query images and key images, respectively, and share the same architecture where an MLP projection head f_w is on top of a backbone network f_θ. Only the query encoder is updated by the gradients backpropagation while the key encoder is updated by the moving average with a momentum. MoCo-v2 maintains an additional dictionary as a queue of features for contrastive learning. Specifically, features in the dictionary are progressively updated. The current mini-chunk features from the key encoder are enqueued and the same number of oldest features are dequeued. MoCo-v2 uses InfoNCE (Oord et al., 2018), a variant of contrastive loss (CL), to maximize the similarity of features from positive pairs and minimize the similarity of features from negative pairs. The contrastive loss is formalized as below.

$$\mathcal{L}_{cl} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{(z_i^\top z_i^+/\tau)}}{e^{(z_i^\top z_i^+/\tau)} + \sum_{z_i^- \in Z^-} e^{(z_i^\top z_i^-/\tau)}}, \tag{1}$$

where N is the number of samples, z_i is the L2-normalized projected feature from the query encoder, z_i^+ is the L2-normalized projected feature of the same input image from the key encoder, Z^- are the negative history features stored in the dictionary and τ is the temperature.

Pre-training. For self-supervised pre-training, we follow the protocol of MoCo-v2 (Chen et al., 2020c), i.e., using the standard ResNet50 backbone (He et al., 2016). The implementation is based on OpenSelfSup[1]. For both joint training and sequential training, the number of training epochs is 200 for each model training, where the convergence of loss is observed. For the instance incremental sequence, we consider one random sequence as the data are randomly divided. While for the random class incremental sequence, distant class incremental sequence, and domain incremental sequence, we experiment with different sequences of data chunks. In particular, considered sequences are obtained through right circular shift operations. For example, if the data sequence length is 4, after splitting all the data into four chunks A, B, C, and D, four sequences, namely A-B-C-D, B-C-D-A, C-D-A-B, and D-A-B-C are used for the sequential pre-training a representation model. The results from different sequences are averaged to obtain the final performance. For comparison, supervised pre-training is also implemented using OpenSelfSup following the recommended training protocol of ImageNet. For supervised pre-training, the classifier layer is reset at a new data chunk.

C Details of Downstream Tasks

We evaluate the transfer performance of the pre-trained models using three different downstream tasks. Following (Chen et al., 2020a), we consider 12 diverse image classification datasets including Food-101 (Bossard et al., 2014), CIFAR10 (Krizhevsky et al., 2009), CIFAR100 (Krizhevsky et al., 2009), Birdsnap (Berg et al., 2014), SUN397 (Xiao et al., 2010), Standard Cars (Krause et al., 2013), FGVC Aircraft (Maji et al., 2013), VOC2007 (Everingham et al., 2010), DTD (Cimpoi et al., 2014), Oxford-IIIT Pets (Parkhi et al., 2012), Caltech-101 (Fei-Fei et al., 2004) and Oxford 102 Flowers (Nilsback & Zisserman, 2008). On these datasets, we evaluate the pre-trained models via the many-shot classification and the few-shot classification (except VOC2007). Both classification protocols are the same as (Ericsson et al., 2021). In addition, we evaluate the pre-trained models on the PASCAL VOC detection task, following the same transfer protocol of MoCo (He et al., 2020). The training data of detection come from VOC2007 and VOC2012, and the test data come from VOC2007.

[1] https://github.com/open-mmlab/OpenSelfSup

Many-shot classification. Many-shot classification is a widely used evaluation protocol (Chen et al., 2020b; He et al., 2020). To evaluate the pre-trained representations, a linear classifier is directly added to the pre-trained feature encoder. During the downstream task evaluation, only the added linear classifier is fine-tuned using the downstream training data while the feature encoder is frozen. In this way, the downstream transfer performance can directly reflect the generalization ability of the pre-trained representation models.

Few-shot classification. Few-shot classification reflects how well the pre-trained models perform on downstream tasks in the few-shot learning regime. Specifically, we consider 5-way 5-shot few-shot tasks on 11 downstream classification datasets, following the few-shot setting in (Ericsson et al., 2021).

Detection. To further evaluate the transferability of the pre-trained models on more downstream scenarios, we consider object detection as a downstream task, where the fine-grained spatial location information is more important, compared with classification tasks. To be specific, we follow the settings in (He et al., 2020), i.e., adopting the Faster-RCNN (Ren et al., 2015) with a backbone of R50-dilated-C5 and fine-tuning all layers including the pre-trained representation network.

We perform no hyper-parameter tuning for few-shot evaluation and detection evaluation. As for the linear evaluation protocol, we adopt the logistic regression and only tune the weight decay value. The inversed weight decay values for all downstream classification datasets are given in Table 5.

Table 5: The inverse of regularization strength (weight decay value) used in many-shot logistic regression evaluation on 12 different downstream classification datasets. SSL models: self-supervised models. SL models: supervised models.

Dataset	SSL Models	SL Models
Aircraft	5623.413277133687	9.99999985098839
Caltech-101	316227.7712565657	0.3162277621819913
Flowers	31622.77530666721	999.999952502551
Pets	999.999952502551	562.3413185099295
Cars	5623.413277133687	17.782794106882072
DTD	1778.2794843157246	0.0177827946252197
Food	177827.94843157247	0.0562341298247638
CIFAR10	316227.7712565657	0.0562341298247638
CIFAR100	100.00000223517424	0.0562341298247638
Birdsnap	1778.27948431572	0.1
SUN397	100.00000223517424	0.0177827946252197
VOC2007	9.99999985098839	0.005623413223739

D Detailed of Continual Learning Methods

Continual learning is assumed to suffer from catastrophic forgetting of previously learned knowledge in supervised learning (Goodfellow et al., 2013; Kirkpatrick et al., 2017; McCloskey & Cohen, 1989), leading to significant performance degradation of previous tasks. Here we introduce the continual learning techniques we adopt, including data replay (Rolnick et al., 2019; Rebuffi et al., 2017) and regularization-based method, e.g., Memory Aware Synapses (MAS) (Aljundi et al., 2018).

Data replay. Data relay is a simple yet effective method for alleviating the catastrophic forgetting problem during the continual learning process. Specifically, we need to maintain a replay buffer and store a selected subset of samples from each learned task in the buffer. Then we just retrain on samples in the replay buffer to revisit old tasks while training the model for a new task.

To perform data replay in the sequential training process, we maintain a replay buffer consisting of images sampled from previous data chunks. After finishing the sequential training with each data chunk, we randomly select 10% data of this chunk and store sampled data in the replay buffer. For the sequential training with the current data chunk, we directly mix current data with data in the replay buffer for self-supervised pre-training e.g. MoCo-v2 training using Eqn. (1).

MAS. Regularization-based methods aim to mitigate the catastrophic forgetting by consolidating previous knowledge with an added regularization term in the loss function. One typical unsupervised

regularization-based method is MAS (Aljundi et al., 2018). Specifically, MAS proposes to compute gradients the squared L2-norm of the encoder output f_θ as the importance weights of parameters.

$$\Omega_{ij} = \frac{1}{N} \sum_{k=1}^{N} \frac{\partial[\|f_\theta(x)\|_2^2)]}{\partial \theta_{ij}}. \tag{2}$$

With the parameter regularization term added, the resulting loss function with the coefficient λ is shown as below.

$$\mathcal{L}(\theta) = \mathcal{L}_{cl}(\theta) + \lambda \sum_{i,j} \Omega_{ij}(\theta_{ij} - \theta_{ij}^*)^2. \tag{3}$$

In MoCo-v2, the query encoder is considered as the representation network, we thus only impose the MAS regularization term on parameters of the query encoder. Specifically, the regularization coefficient λ is fixed to be 100. Following the prevailing use of MAS regularization (Aljundi et al., 2018; 2019), we update the MAS importance weights Ω_{ij} to cover information of each data chunk in the sequential training process. To be specific, after finishing the sequential training with each data chunk, we leverage both the current data chunk and data in the replay buffer to estimate importance weights for the trained model using Eqn. (2). Then we update the stored sequential importance weights by a cumulative moving average of current and previous estimated importance weights, following (Aljundi et al., 2019). As for the model training on the current data chunk, we apply the parameter regularization using previous importance weights and optimize the model using Eqn. (3).

To sum up, besides the sequentially trained model, both above methods require extra storage for sequential self-supervised pre-training. For the 10% data replay method, we need to only keep 10% data of each previous data chunk for sequential training. For the MAS regularization method, we only require to save a set of importance weights for the model and then update the importance weights sequentially.

E Details of CKA Similarity

To further understand the sequential self-supervised pre-training, we then take a closer look at the learned feature representations during the sequential training process. We leverage the linear centered kernel alignment (CKA) (Kornblith et al., 2019a) to measure the similarity of output features between two different representation networks given the same data set as input. If we consider the size of the data set as n and the feature dimension for two networks as d_1 and d_2, respectively. We use the selected data set to extract features $X \in \mathbb{R}^{n \times d_1}$ from one representation network and features $Y \in \mathbb{R}^{n \times d_2}$ from another representation network. In our experiments, n is 50,000 and both d_1 and d_2 are 2,048. We first preprocess the two representation matrices by centering the columns. Then the linear CKA similarity between two representations X and Y can be computed as below:

$$CKA(X, Y) = \frac{\|X^T Y\|_F^2}{\|X^T X\|_F^2 \|Y^T Y\|_F^2}.$$

F Results of instance incremental sequence

Transfer learning results of self-supervised pre-training with the instance incremental sequence are evaluated on all three downstream tasks. For results of both many-shot classification and few-shot classification in Figure 11, we find sequential SSL performs comparably with joint SSL on all downstream datasets, with the average performance gap between sequential training and joint training less than 1%, while there exists evident gaps, more than 4%, between sequential supervised learning and joint supervised learning.

G Results of BYOL

To evaluate whether sequential training performs well for other SSL methods, we conduct the challenging distant class incremental sequence experiments with BYOL (Grill et al., 2020). The results of BYOL are shown in Figure 12. Similar to the observations with MoCo-v2 in Section 4.1, sequential SSL is visibly inferior to joint SSL on streaming data with severe distribution shift, but

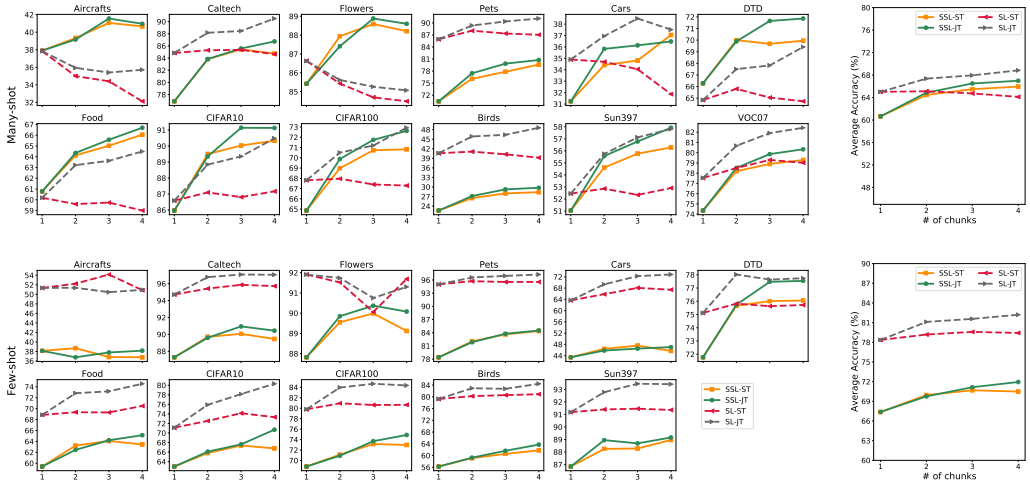


Figure 11: Linear and few-shot evaluation results of **instance incremental sequence**. on the left are the results of each dataset. On the right are averaged results across all left datasets.

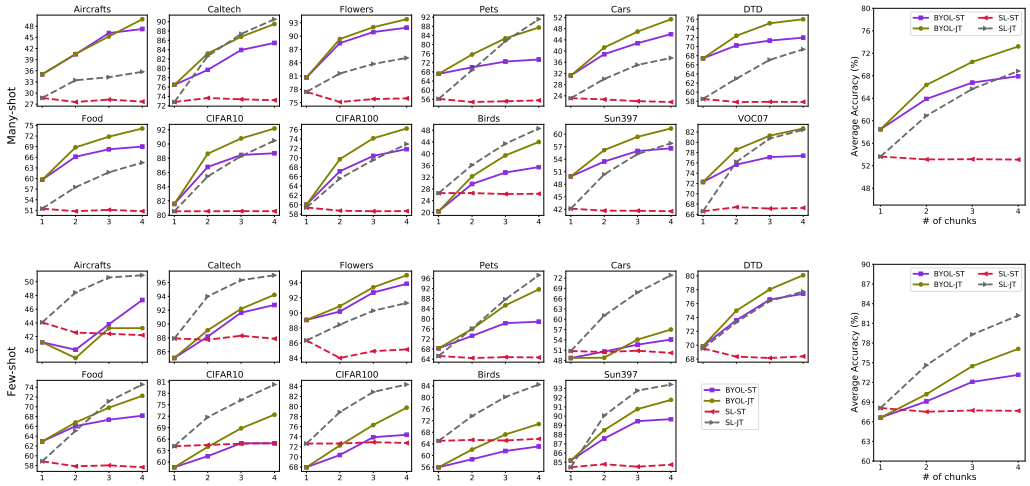


Figure 12: Linear and few-shot evaluation results of **distant incremental sequence** for BYOL. on the left are the results of each dataset. On the right are averaged results across all left datasets.

sequential SL performs obviously worse than joint SL. In addition, compared with SL models, SSL models show significantly smaller performance gaps between sequential training and joint training.