CVPR 2020 continual learning in computer vision competition: Approaches, results, current challenges and future directions

(Article begins on next page)

18 September 2024

CVPR 2020 Continual Learning in Computer Vision Competition: Approaches, Results, Current Challenges and Future Directions

Vincenzo Lomonaco [1 2] **Lorenzo Pellegrini** [1] **Pau Rodriguez** [3] **Massimo Caccia** [4 3] **Qi She** [5 2] **Yu Chen** [6]
Quentin Jodelet [7 8] **Ruiping Wang** [9] **Zheda Mai** [10] **David Vazquez** [3] **German I. Parisi** [11 2] **Nikhil Churamani** [12]
Marc Pickett [13] **Issam Laradji** [3] **Davide Maltoni** [1]

Abstract

In the last few years, we have witnessed a renewed and fast-growing interest in continual learning with deep neural networks with the shared objective of making current AI systems more adaptive, efficient and autonomous. However, despite the significant and undoubted progress of the field in addressing the issue of catastrophic forgetting, benchmarking different continual learning approaches is a difficult task by itself. In fact, given the proliferation of different settings, training and evaluation protocols, metrics and nomenclature, it is often tricky to properly characterize a continual learning algorithm, relate it to other solutions and gauge its real-world applicability. The first Continual Learning in Computer Vision challenge held at CVPR in 2020 has been one of the first opportunities to evaluate different continual learning algorithms on a common hardware with a large set of shared evaluation metrics and 3 different settings based on the realistic CORe50 video benchmark. In this paper, we report the main results of the competition, which counted more than 79 teams registered, 11 finalists and 2300$ in prizes. We also summarize the winning approaches, current challenges and future research directions.

1. Introduction

Continual Learning, the new deep learning embodiment of a long-standing paradigm in machine learning research and
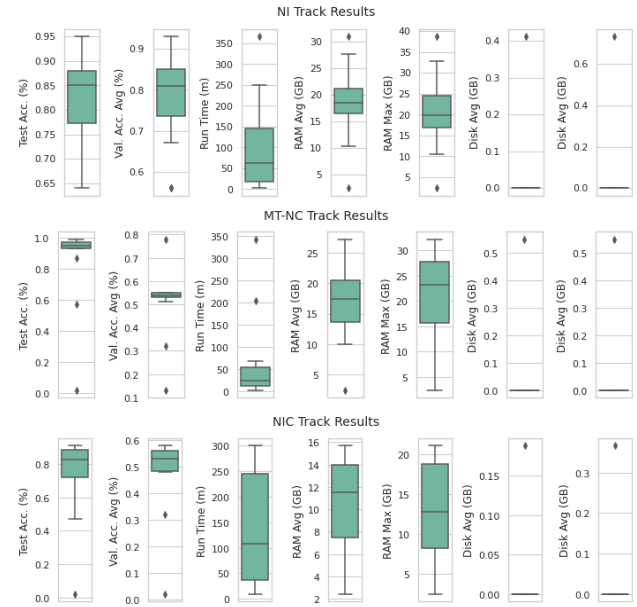

Figure 1. Results distributions for the three tracks (NI, MT-NC and NIC) across the 11 finalists solutions and the main evaluation metrics used for the competition: total test accuracy (%) at the end of the training, average validation accuracy over time (%), maximum and average RAM/Disk usage (GB).

AI also known as Incremental or Lifelong Learning, has received a renewed attention from the research community over the last few years (Parisi et al., 2019; Lomonaco, 2019; Lesort et al., 2020). Continual learning, indeed, appears more and more clearly as the only viable option for sustainable AI agents that can scale efficiently in terms of general intelligence capabilities while adapting to ever-changing environments and unpredictable circumstances over time. Even not considering long-term goals of truly intelligent AI agents, from a pure engineering perspective, continual learning is a very desirable option for any AI technology learning on premises or at the edge on embedded devices without the need of moving private data to remote cloud infrastructures (Farquhar & Gal, 2019).

[1]University of Bologna [2]ContinualAI Research [3]Element AI [4]MILA [5]ByteDance AI Labs [6]University of Bristol [7]Tokyo Institute of Technology [8]AIST RWBC-OIL [9]Chinese Academy of Sciences [10]University of Toronto [11]University of Hamburg [12]University of Cambridge [13]Google AI. Correspondence to: Vincenzo Lomonaco <vincenzo.lomonaco@unibo.it>.

However, gradient-based architectures, such as neural networks trained with Stochastic Gradient Descent (SGD), notably suffer from catastrophic forgetting or interference (McCloskey & Cohen, 1989; Robins, 1995; French, 1999), where the network parameters are rapidly overwritten when learning over non-stationary data distributions to model only the most recent. In the last few years, significant progresses have been made to tame the issue. Nevertheless, comparing continual learning algorithms today constitutes a hard task (Díaz-Rodríguez et al., 2018). This is mainly due to the proliferation of different settings only covering partial aspects of the continual learning paradigm, with diverse training and evaluation protocols, metrics and datasets used (Lesort et al., 2020; Caccia et al., 2020). Another important question is whether such algorithms, that have mostly been proved on artificial benchmarks such as MNIST (LeCun et al., 1998) or CIFAR (Krizhevsky et al., 2009), can scale and generalize to different settings and real-world applications.

The *1st Continual Learning in Computer Vision Challenge*, organized within the *CLVision* workshop at CVPR 2020, is one of the first attempts to address these questions. In particular, the main objectives of the competition were:

- Invite the research community to scale up continual learning approaches to natural images and possibly on video benchmarks.

- Invite the community to work on solutions that can generalize over multiple continual learning protocols and settings (e.g. with or without a "task" supervised signal).

- Provide the first opportunity for a comprehensive evaluation on a shared hardware platform for a fair comparison.

Notable competitions previously organized in this area include: the Pascal 2 EU network of excellence challenge on *"covariate shift"*, organized in 2005 (Quionero-Candela et al., 2005; 2009); the *Autonomous Lifelong Machine Learning with Drift* challenge organized at NeurIPS 2018 (Escalante et al., 2020) and the *IROS 2019 Lifelong Robotic Vision* challenge (Bae et al., 2020). While the first two competitions can be considered as the first continual learning challenges ever organized, they were based on low-dimensional features benchmarks that made it difficult to understand the scalability of the proposed methods to more complex settings with deep learning based techniques. The latest competition, instead, has been one of the first challenges organized within robotic vision realistic settings. However, it lacked a general focus on computer vision applications as well as a comprehensive evaluation on 3 different settings and 4 tracks.
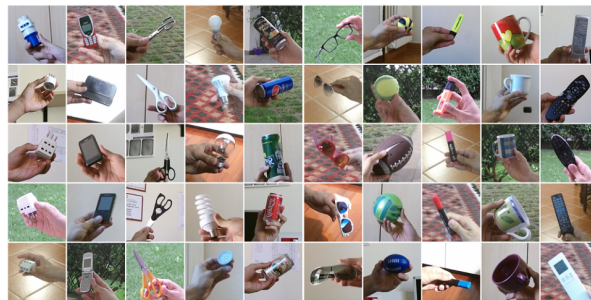

Figure 2. Example images of the 50 objects in *CORe50*, the main video dataset used in the challenge. Each column denotes one of the 10 categories (Lomonaco & Maltoni, 2017a).

For transparency and reproducibility, we openly release the finalists' dockerized solutions as well as the initial baselines at the following link: `https://github.com/vlomonaco/cvpr_clvision_challenge`.

2. Competition

The *CLVision competition* was planned as a 2-phase event (pre-selection and finals), with 4 tracks and held online from the *15th of February 2020* to the *14th of June 2020*. The *pre-selection* phase, based on the codalab online evaluation framework[1], lasted 78 days and was followed by the finals where a dockerized solution had to be submitted for remote evaluation on a shared hardware. In the following section, the dataset, the different tracks, the evaluation metric used and the main rules of the competition are reported in detail. Finally, the main competition statistics, participants and winners are presented.

2.1. Dataset

CORe50 (Lomonaco & Maltoni, 2017b) was specifically designed as an object recognition video benchmark for continual learning. It consists of 164,866 128×128 images of 50 domestic objects belonging to 10 categories (see Figure 2); for each object the dataset includes 11 video sessions (∼300 frames recorded with a Kinect 2 at 20 fps) characterized by relevant variations in terms of lighting, background, pose and occlusions. Classification on CORe50 can be performed at Object level (50 classes) or at Catagory level (10 classes). The former, being a more challenging task, was the configuration chosen for this competition. The egocentric vision of hand-held objects allows for the emulation of a scenario where a robot has to incrementally learn to recognize objects while manipulating them. Objects are presented to the robot by a human operator who can also provide the labels, thus enabling a supervised classification

---

[1] `https://codalab.org`

CVPR 2020 Continual Learning in Computer Vision Competition

Table 1. 11 finalists of the CLVision Competition.

Team Name	Team Members
HaoranZhu	Haoran Zhu
ICT_VIPL	Chen He, Qiyang Wan, Fengyuan Yang, Ruiping Wang, Shiguang Shan, Xilin Chen
JimiB	Giacomo Bonato, Francesco Lakj, Alex Torcinovich, Alessandro Casella
Jodelet	Quentin Jodelet, Vincent Gripon, Tsuyoshi Murata
Jun2Tong	Junyong Tong, Amir Nazemi, Mohammad Javad Shafiee, Paul Fieguth
MrGranddy	Vahit Bugra Yesilkaynak, Firat Oncel, Furkan Ozcelik, Yusuf Huseyin Sahin, Gozde Unal
Noobmaster	Zhaoyang Wu, Yilin Shao, Jiaxuan Zhao, and Bingnan Hu
Sahinyu	Yusuf H. Sahin, Furkan Ozcelik, Firat Oncel, Vahit Bugra Yesilkaynak, Gozde Unal
Soony	Soonyong Song, Heechul Bae, Hyonyoung Han, Youngsung Son
UT_LG	Zheda Mai, Hyunwoo Kim, Jihwan Jeong, Scott Sanner
YC14600	Yu Chen, Jian Ma, Hanyuan Wang, Yuhang Ming, Jordan Massiah, Tom Diethe

(such an applicative scenario is well described in Pasquale et al. (2019); She et al. (2020)).

2.2. Tracks

Based on the CORe50 dataset, the challenge included four different tracks based on the different settings considered:

1. New Instances (NI): In this setting 8 training batches of the same 50 classes are encountered over time. Each training batch is composed of different images collected in different environmental conditions.

2. Multi-Task New Classes (MT-NC)[2]: In this setting the 50 different classes are split into 9 different tasks: 10 classes in the first batch and 5 classes in the other 8. In this case the task label will be provided during training and test.

3. New Instances and Classes (NIC): this protocol is composed of 391 training batches containing 300 images of a single class. No task label will be provided and each batch may contain images of a class seen before as well as a completely new class.

4. All together (ALL): All the settings presented above.

Each participant of the challenge could choose in which of the main three tracks (NI, MT-NC, NIC) to compete. Those participants that decided to participate to all the three main tracks were automatically included in the ALL track as well, the most difficult and ambitious track of the competition.

2.3. Evaluation Metric

In the last few years the main evaluation focus in continual learning has always been centered around accuracy-related

forgetting metrics. However, as argued by Díaz-Rodríguez et al. (2018), this may lead to biased conclusion not accounting for the real scalability of such techniques over an increasing number of tasks/batches and more complex settings. For this reason, in the competition each solution was evaluated across a number of metrics:

1. *Final accuracy on the test set*[3]: computed only at the end of the training procedure.

2. *Average accuracy over time on the validation set*: computed at every batch/task.

3. *Total training/test time*: total running time from start to end of the main function (in minutes).

4. *RAM usage*: total memory occupation of the process and its eventual sub-processes. It is computed at every epoch (in MB).

5. *Disk usage*: only of additional data produced during training (like replay patterns) and additionally stored parameters. It is computed at every epoch (in MB).

The final aggregation metric (CL_{score}) is the weighted average of the 1-5 metrics (0.3, 0.1, 0.15, 0.125, 0.125 respectively).

2.4. Rules and Evaluation Infrastructure

In order to provide a fair evaluation while not constraining each participants to simplistic solutions due to a limited server-side computational budget, the challenge was based on the following rules:

1. The challenge was based on the *Codalab* platform. For the pre-selection phase, each team was asked to run the experiments *locally* on their machines with the

---

[2]Multi-Task-NC constitutes a simplified variation of the originally proposed New Classes (NC) protocol (Lomonaco & Maltoni, 2017b) (where the task label is not provided during train and test).

[3]Accuracy in CORe50 is computed on a fixed test set. Rationale behind this choice is explained in (Lomonaco & Maltoni, 2017b)

help of a Python repository to easily load the data and generate the submission file (with all the necessary data to execute the submission remotely and verify the adherence to the competition rules if needed). The submission file, once uploaded, was used to compute the CL_{Score} which determined the ranking in each scoreboard (one for each track).

2. It was possible to optimize the data loader, but not to change the data order or the protocol itself.

3. The top 11 teams in the scoreboard at the end of the pre-selection phase were selected for the final evaluation.

4. The final evaluation consisted in a *remote* evaluation of the final submission for each team. This is to make sure the final ranking was computed in the same computational environment for a fair comparison. In this phase, experiments were run remotely for all the teams over a 32 CPU cores, 1 NVIDIA Titan X GPU, 64 GB RAM Linux system. The max running time was capped at 5 hours for each submission/track.

5. Each team selected for the final evaluation had to submit a single dockerized solution which had to contain the exact same solution submitted for the last co-dalab evaluation. The initial docker image (provided in the initial challenge repository) could have been customized at will but without exceeding 5 GB.

It is worth noting that only the test accuracy was considered in the ranking of the pre-selection phase of the challenge, since the evaluation was run on participants' local hardware. However, since it was not possible to submit a different solution for the final evaluation, this ensured the competition was not biased on the sole accuracy metric.

The financial budget for the challenge was entirely allocated for the monetary prizes in order to stimulate participation:

- 800$ for the participant with highest average score accross the three tracks (e.g the ALL track).

- 500$ for the participant with highest score on the NI track.

- 500$ for the participant with highest score on the MT-NC track.

- 500$ for the participant with highest score on the NIC track.

These prizes were kindly sponsored by *Intel Labs (China)*, while the remote evaluation was performed thanks to the hardware provided by the *University of Bologna*.

2.5. Participants and Finalists

The challenge counted the participation of 79 teams worldwide that competed during the pre-selection phase. From those 79 teams only 11 qualified to the finals with a total of 46 people involved and an average team components number of 4. In Table 1 the 11 finalist teams and their members are reported.

3. Continual Learning Approaches

In this section we discuss the baselines made available as well as the continual learning approaches of the winning teams in more details. On the official competition website an extended report for each of the finalist team detailing their approach is also publicly available.[4]

3.1. Baselines

In order to better understand the challenge complexity and the competitiveness of the proposed solutions, three main baselines were included for each of the 4 tracks:

- *Naive*: This is the basic *finetuning* strategy, where the standard SGD optimization process is continued on the new batches/tasks without any additional regularization constraint, architectural adjustment or memory replay process.

- *Rehearsal*: In this baseline the *Naive* approach is augmented with a basic replay process with a growing external memory, where 20 images for each batch are stored.

- *AR1* with Latent Replay*: a recently proposed strategy (Pellegrini et al., 2020) showing competitive results on CORe50 with a shared, non fine-tuned hyper-parametrization across the three main tracks.

3.2. Team ICT_VIPL

General techniques for all tracks. To improve their performance the ICT_VIPL team used: (1) *Heavy Augmentation* with the Python `imgaug` library[5]; (2) resize the input image to $224{\times}224$ to encourage more knowledge transfer from the ImageNet pretrained model; (3) employ an additional exemplar memory for episodic memory replay to alleviate catastrophic forgetting (randomly select $2 \sim 3\%$ of the training samples); (4) striking a balance between performance and model capacity by using a moderately deep network ResNet-50. As for efficiency, they leveraged the PyTorch Dataloader module for multi-thread speed-up.

---

[4]https://sites.google.com/view/clvision2020/challenge
[5]https://imgaug.readthedocs.io

Special techniques for individual tracks. For NI track, there is no special design over the general techniques above and they only tune the best hyper-parameters. For Multi-Task-NC track, they carefully design a pipeline that disentangles representation and classifier learning, which shows very high accuracy and the pipeline is as below (D_i is the set of exemplars for Task i and $|D_i|$ is its size):

For Task 0: (1) Train the feature extractor $f(x)$ and the first head $c_0(z)$ with all training samples; (2) Select N samples randomly and store them in the exemplar memory ($|D_0| = N$).

For Task i ($i = 1, 2, \ldots, 8$): (1) Train head $c_i(z)$ with all training samples of Task i; (2) Drop some samples randomly from the previous memory, keep $|D_j| = \frac{N}{i+1}$ (for all $j < i$); (3) Select $\frac{N}{i+1}$ samples from Task i randomly and store them in the exemplar memory ($|D_i| = \frac{N}{i+1}$); (4) Fine-tune the feature extractor $f(x)$ with all samples in the memory $\cup_j D_j (j \leq i)$. (since the feature extractor alone cannot classify images, a temporary head $c(z)$ is used for training); (5) Fine-tune each head $c_j(z)$ with the corresponding samples in the memory $D_j (j \leq i)$.

For NIC track, based on the assumption that the neural network estimates Bayesian a posteriori probibilities (Richard & Lippmann, 1991), the network outputs are divided by the prior probability for each class inspired by the trick that handles class imbalance (Buda et al., 2018). Such a technique can prevent the classifier from biasing minority class (predict to newly added classes) especially in the first few increments.

3.3. Team Jodelet

The proposed solution consists in the concatenation of a pre-trained deep convolutional neural network used as a feature extractor and an online trained logistic regression combined with a small reservoir memory (Chaudhry et al., 2019) used for rehearsal.

Since the guiding principle of the proposed solution is to limit as much as possible the computational complexity, the model is trained in an online continual learning setting: each training example is only used once. In order to further decrease the memory and computational complexity of the solution at the cost of a slight decrease of the accuracy, the pre-trained feature extractor is fixed and is not fine-tuned during the training procedure. As a result, it is not necessary to apply the gradient descent algorithm to the large feature extractor and the produced representation is fixed. Therefore, it is possible to store the feature representation in the reservoir memory instead of the whole input raw image. In addition to the memory gain, this implies that the replay patterns do not have to go through the feature extractor again, effectively decreasing the computational complexity of the proposed solution.

Among the different architectures and training procedures considered for the feature extractor, ResNet-50 (He et al., 2016) trained by Facebook AI using the Semi-Weakly Supervised Learning procedure (Yalniz et al., 2019) was selected. This training procedure relies on the use of a teacher model and 940 million public images in addition to the ImageNet dataset (Russakovsky et al., 2015). Compared with the reference training procedure in which the feature extractor is solely trained on the ImageNet dataset, this novel training procedure allows for a consequent increase of the accuracy without modifying the architecture: while the difference of Top-1 accuracy between both training procedures for ResNet-50 is about 5.0% on Imagenet, the difference increases up to 11.1% on the NIC track of the challenge. Moreover, it should be noted that on the three tracks of the challenge, ResNet-18 feature extractor trained using this new procedure is able to reach an accuracy comparable with the one of the reference ResNet-50 feature extractor trained only on ImageNet, while being considerably smaller and faster.

For reasons of consistency, the same hyperparameters have been used for the three tracks of the challenge and have been selected using a grid search.

3.4. Team UT_LG

Batch-level Experience Replay with Review In most *Experience Replay* based methods, the incoming mini-batch is concatenated with another mini-batch of samples retrieved from the memory buffer. Then, they simply take an SGD step with the concatenated samples, followed by an update of the memory (Chaudhry et al., 2019; Caccia et al., 2019). Team UT_LG method makes two modifications. Firstly, to reduce the number of retrieval and update steps, they concatenate the memory examples at the batch level instead of at the mini-batch level. Concretely, for every epoch, they draw a batch of data $D_{\mathcal{M}}$ randomly from memory with size $replay\_sz$, concatenate it with the current batch and conduct the gradient descent parameters update. Moreover, they add a review step before the final testing, where they draw a batch of size D_R from memory and conduct the gradient update again. To prevent overfitting, the learning rate in the review step is usually lower than the learning rate used when processing incoming batches. The overall training procedure is presented in Algorithm 1.

Data Preprocessing (1) Centering-cropping the image with a (100, 100) window to make the target object occupy more pixels in the image. (2) Resizing the cropped image to (224, 224) to ensure no size discrepancy between the input of the pre-trained model and the training images. (3) Pixel-level and spatial-level data augmentation to improve

generalization. The details of their implementation can be found in (Mai et al., 2020)

Algorithm 1 Batch-level Experience Replay with Review

procedure BERR($\mathcal{D}$, mem_sz, replay_sz, review_sz, lr_replay, lr_review)

$\quad \mathcal{M} \leftarrow \{\} * mem\_sz$
$\quad$ **for** $t \in \{1, \dots, T\}$ **do**
$\quad\quad$ **for** epochs **do**
$\quad\quad\quad$ **if** $t > 1$ **then**
$\quad\quad\quad\quad D_{\mathcal{M}} \overset{\text{replay\_sz}}{\sim} \mathcal{M}$
$\quad\quad\quad\quad D_{\text{train}} = D_{\mathcal{M}} \cup D_t$
$\quad\quad\quad$ **else**
$\quad\quad\quad\quad D_{\text{train}} = D_t$
$\quad\quad\quad \theta \leftarrow \text{SGD}(D_{\text{train}}, \theta, \text{lr\_replay})$
$\quad\quad \mathcal{M} \leftarrow UpdateMemory(D_t, \mathcal{M}, \text{mem\_sz})$
$\quad D_R \overset{\text{review\_sz}}{\sim} \mathcal{M}$
$\quad \theta \leftarrow \text{SGD}(D_R, \theta, \text{lr\_review})$
$\quad$ **return** θ

3.5. Team Yc14600

The use of episodic memories in continual learning is an efficient way to prevent the phenomenon of *catastrophic forgetting*. In recent studies, several gradient-based approaches have been developed to make more efficient use of compact episodic memories. The essential idea is to use gradients produced by samples from episodic memories to constrain the gradients produced by new samples, *e.g.* by ensuring the inner product of the pair of gradients is non-negative (Lopez-Paz & Ranzato, 2017) as follows:

$$\langle g_t, g_k \rangle = \left\langle \frac{\partial \mathcal{L}(x_t, \theta)}{\partial \theta}, \frac{\partial \mathcal{L}(x_k, \theta)}{\partial \theta} \right\rangle \geq 0, \forall k < t \quad (1)$$

where t and k are time indices, x_t denotes a new sample from the current task, and x_k denotes a sample from the episodic memory. Thus, the updates of parameters are forced to preserve the performance on previous tasks as much as possible. Equation (1) indicates larger cosine similarities between gradients produced by current and previous tasks result in improved generalisation. This in turn indicates that samples that lead to the most diverse gradients provide the most difficulty during learning.

Through empirical studies the team members found that the discrimination ability of representations strongly correlates with the diversity of gradients, and more discriminative representations lead to more consistent gradients. They use this insight to introduce an extra objective Discriminative Representation Loss (DRL) into the optimization objective of classification tasks in continual learning. Instead of explicitly refining gradients during training process, DRL helps

with decreasing gradient diversity by optimizing the representations. As defined in Equation (2), DRL consists of two parts: one is for minimizing the similarities of representations between samples from different classes ($\mathcal{L}_{bt}$), the other is for minimizing the similarities of representations between samples from a same class ($\mathcal{L}_{wi}$) for preserving information of representations for future tasks.

$$\min_{\Theta} \mathcal{L}_{DR} = \min_{\Theta}(\mathcal{L}_{bt} + \mathcal{L}_{wi}),$$

$$\mathcal{L}_{bt} = \frac{1}{B_{bt}} \sum_{l=1}^{L} \sum_{i=1}^{B} \sum_{j=1, y_j \neq y_i}^{B} \langle h_{l,i}, h_{l,j} \rangle, \quad (2)$$

$$\mathcal{L}_{wi} = \frac{1}{B_{wi}} \sum_{l=1}^{L} \sum_{i=1}^{B} \sum_{j=1, j \neq i, y_j = y_i}^{B} \langle h_{l,i}, h_{l,j} \rangle.$$

where Θ denotes the parameters of the model, L is the number of layers of the model, B is training batch size. B_{bt} and B_{wi} denote the number of pairs of samples in the training batch that are from different classes and the same class, respectively, $h_{l,i}$ is the output of layer l by input x_i and y_i is the label of x_i. Please refer to (Chen et al., 2020) for more details.
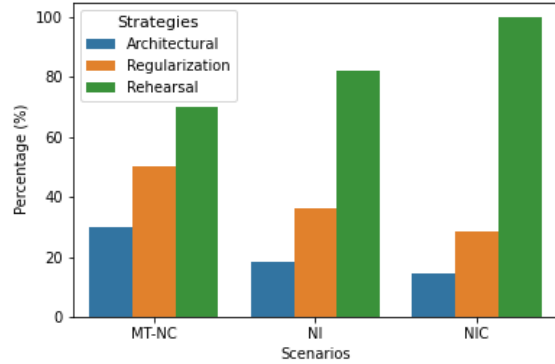
Figure 3. Percentage (%) of finalists solutions for each track employing an *architectural*, *regularization* or *rehearsal* strategy. Percentages do not sum to 100% since many approached used hybrid strategies. Better viewed in colors.

4. Competition Results

In this section we detail the main results of the competition for each of the main three tracks (NI, MT-NC & NIC) as well as the averaged track ALL, which determined the overall winner of the challenge. For each track the teams are ranked as follows: i) each metric is normalized across between 0 and 1; ii) the CL_{score} is computed as a weighted average; ii) results are ordered in descending order.

In the next sections we report the results with their absolute values to better grasp the quality of the solutions proposed

and their portability in different applicative contexts.

4.1. New Instances (NI) Track

In Tab. 2 the main results for the *New Instances* (NI) track are reported. In Tab. 6, additional details (not taken into account for the evaluation) for each solution are shown. In this track, the *UT_LG* obtained the best CL_{Score} with a small gap w.r.t. its competitors. The test accuracy tops 91% for the winning team, showing competitive performance also in real-world non-stationary applications. It is worth noting that the top-4 solutions all employed a rehearsal-based technique, only in one case supported by a regularization counterpart.

4.2. Multi-Task NC (MT-NC) Track

For the MT-NC track, results are reported in Tab. 3 and additional details in Tab. 7 of the Appendix. In this scenario, arguably the easiest since it provided an additional supervised signal (the Task label) the AR1 baseline resulted as the best scoring solution. In fact, while achieving lower accuracy results than the other top-7 solutions, it offered a more efficient algorithmic proposal in terms of both memory and computation (even without a careful hyper-parametrization). It is also interesting to note that, in this scenario, it is possible to achieve impressive accuracy performance ($\sim$99%) within reasonable computation and memory constraints as shown by the ICT_VIPL team, the only solution who opted for a disk-based exemplars memorization.

4.3. New Instances (NIC) Track

The NIC track results are reported in Tab. 4. Additional details of each solution are also made available in Tab. 8. Only 7 over 11 finalist teams submitted a solution for this track. In this case, it is possible to observe generally lower accuracy results and an increase in the running times across the 391 batches.

4.4. All (ALL) Track

Finally in Tab. 5 the results averaged across tracks are reported for the ALL scoreboard. Also in this case the competing teams were 7 over a total of 11 with *UT_LG* as the winning team. With an average testing accuracy of $\sim$92%, a average memory consumption of $\sim$10 GB and a running time of $\sim$68 minutes, its relatively simple solution suggests continual learning for practical object recognition applications to be feasible in the real-world, even with a large number of small non-i.i.d. bathes.

4.5. Discussion

Given the main competition results and the additional solutions details reported in Appendix A, we can formulate a number of observations to better understand current issues, consolidated approaches and possible future directions for competitive continual learning algorithms tested on real-world computer vision applications.

In particular, we note:

- *Different difficulty for different scenarios*: averaging the 11 finalists test accuracy results we can easily deduce that the MT-NC track or scenario was easier than the NI one ($\sim$85% vs $\sim$82%), while the NIC track was the most difficult with a average accuracy of $\sim$72%. This is not totally surprising, considering that the MT-NC setting allows access to the additional task labels and the NI scenario does not include dramatic distributional shifts, while the NIC one includes a substantially larger number of smaller training batches. Moreover, a number of researchers already pointed out how different training/testing regimes impacts forgetting and the continual learning process (Mirzadeh et al., 2020; Maltoni & Lomonaco, 2019; Hayes et al., 2018).

- *100% of the teams used a pre-trained model*: All the solutions, for all the tracks started from a pre-trained model on ImageNet. While starting from a pre-trained model is notably becoming a standard for real-world computer vision applications, we find it interesting to point out such a pervasive use in the challenge. While this does not mean pre-trained model should be used for every continual learning algorithm in general, it strongly suggests that for solving real-world computer vision application today, pre-training is mostly needed.

- $\sim$*90% of the teams used a rehearsal strategy*: rehearsal constitutes today one of the easiest and effective solution to continual learning where previous works (Hayes et al., 2019) have shown that even a very small percentage of previously encountered training data can have huge impacts on the final accuracy performance. Hence, it is not surprising that a large number of teams opted to use it for maximizing the CL_{score}, which only slightly penalized its usage.

- $\sim$*45% of the teams used a regularization approach*: regularization strategies have been extensively used in the competition. It worth noting though, that only 1 team used it alone and not in conjunction with a plain rehearsal or architectural approaches.

- *only $\sim$27% of the teams used an architectural approach*: less then one third of the participants did use an architectural approach but only on conjunction with a rehearsal or regularization one. This evidence reinforces the hypothesis that architectural-only approaches are difficult to scale efficiently over a large number of tasks or batches (Rusu et al., 2016).

Table 2. NI track results for the 11 finalists of the competition and the three baselines.

TEAM NAME	TEST ACC (%)	VAL ACC$_{avg}$ (%)	RUN$_{time}$ (M)	RAM$_{avg}$ (MB)	RAM$_{max}$ (MB)	DISK$_{avg}$ (MB)	DISK$_{max}$ (MB)	CL_{score}
UT_LG	0.91	0.90	63.78	11429.83	11643.63	0	0	0.692
Yc14600	0.88	0.85	22.58	17336.38	18446.90	0	0	0.648
ICT_VIPL	0.95	0.93	113.70	2459.42	2460.16	421.875	750	0.629
Jodelet	0.84	0.85	3.11	18805.60	18829.96	0	0	0.612
Soony	0.85	0.81	25.57	16662.73	17000.10	0	0	0.602
JimiB	0.91	0.89	248.82	19110.84	25767.74	0	0	0.573
Jun2tong	0.84	0.76	62.48	20968.43	23252.39	0	0	0.550
Sahinyu	0.88	0.81	156.64	26229.77	32176.76	0	0	0.538
Ar1	0.75	0.73	17.18	10550.61	10838.79	0	0	0.520
Noobmaster	0.85	0.75	74.54	31750.19	39627.31	0	0	0.504
MrGranddy	0.88	0.84	249.28	28384.06	33636.52	0	0	0.501
Naïve	0.66	0.56	2.61	18809.50	18830.11	0	0	0.349
Rehearsal	0.64	0.56	3.79	21685.03	21704.76	0	0	0.326
HaoranZhu	0.70	0.67	366.22	21646.78	21688.30	0	0	0.263
AVG	0.82	0.78	100.74	18987.80	21135.96	30.13	53.57	0.52

Table 3. NC track results for the 11 finalists of the competition and the three baselines. Teams not appearing in the table did not compete in this track.

TEAM NAME	TEST ACC (%)	VAL ACC$_{avg}$ (%)	RUN$_{time}$ (M)	RAM$_{avg}$ (MB)	RAM$_{max}$ (MB)	DISK$_{avg}$ (MB)	DISK$_{max}$ (MB)	CL_{score}
Ar1	0.93	0.53	16.02	10263.19	14971.72	0	0	0.693
UT_LG	0.95	0.55	19.02	13793.31	16095.20	0	0	0.691
Yc14600	0.97	0.54	11.81	15870.62	19403.57	0	0	0.686
Soony	0.97	0.55	55.02	14005.91	16049.12	0	0	0.679
Jodelet	0.97	0.55	2.55	17893.58	23728.84	0	0	0.679
Jun2tong	0.96	0.55	28.80	18488.68	19588.57	0	0	0.671
ICT_VIPL	0.99	0.55	25.20	2432.56	2432.84	562.5	562.5	0.630
Rehearsal	0.87	0.51	4.49	20446.93	28329.14	0	0	0.626
JimiB	0.95	0.78	204.56	21002.95	24528.27	0	0	0.607
MrGranddy	0.94	0.54	46.52	27904.55	32921.94	0	0	0.604
Noobmaster	0.95	0.53	68.07	27899.86	32910.23	0	0	0.597
HaoranZhu	0.57	0.32	343.50	21223.30	28366.48	0	0	0.351
Naïve	0.02	0.13	3.41	17897.38	23726.40	0	0	0.318
AVG	0.85	0.51	63.77	17624.83	21773.26	43.27	43.27	0.60

- *Increasing replay usage with track complexity*: as shown in Fig. 3, it is worth noting that as the track complexity increased, the proposed solutions tended to include more replay mechanisms. For example, for the NIC track, all the approaches included rehearsal, often used in conjunction with a regularization or architectural approach.

- *High memory replay size*: it is interesting to note that many CL solutions employing rehearsal have chosen to use a *growing* memory replay buffer rather than a fixed one with an average maximum memory size (across teams and tracks) of ∼26k patterns. This is a very large number considering that is about ∼21% of the total CORe50 training set images.

- *Different hyper-parameters selection*: An important

note to make is about the hyperparameters selection and its implication to algorithms *generalization* and *robustness*. Almost all participants' solutions involved a carefully fine-tuned hyper-parameters selection which was different based on the continual scenario tackled. This somehow highlights the weakness of state-of-the-art algorithms and their inability to truly generalize to novel situations never encountered before. A notably exception is the *AR1* baseline, which performed reasonably well in all the tracks with a shared hyper-parametrization.

5. Conclusions and Future Improvements

The *1st Continual Learning for Computer Vision Challenge* held at CVPR2020 has been one of the first large-scale

Table 4. NIC track results for the 11 finalists of the competition and the three baselines. Teams not appearing in the table did not compete in this track.

TEAM NAME	TEST ACC (%)	VAL ACC$_{avg}$ (%)	RUN$_{time}$ (M)	RAM$_{avg}$ (MB)	RAM$_{max}$ (MB)	DISK$_{avg}$ (MB)	DISK$_{max}$ (MB)	CL_{score}
UT_LG	0.91	0.58	123.22	6706.61	7135.77	0	0	0.706
JODELET	0.83	0.54	14.12	10576.67	11949.16	0	0	0.694
AR1	0.71	0.48	28.19	3307.62	4467.64	0	0	0.693
ICT_VIPL	0.90	0.56	91.29	2485.95	2486.03	192.187	375	0.625
YC14600	0.89	0.57	160.24	16069.91	21550.97	0	0	0.586
REHEARSAL	0.74	0.50	60.32	15038.34	19488.43	0	0	0.585
SOONY	0.82	0.52	280.39	12933.28	14241.57	0	0	0.533
JIMIB	0.87	0.56	272.98	13873.04	21000.51	0	0	0.533
NOOBMASTER	0.47	0.32	300.15	14492.13	18262.32	0	0	0.346
NAÏVE	0.02	0.02	9.45	10583.50	11917.55	0	0	0.331
AVG	0.72	0.47	134.03	10606.70	13249.99	19.22	37.50	0.56

Table 5. ALL track results for the 11 finalists of the competition and the three baselines. Teams not appearing in the table did not compete in this track.

TEAM NAME	TEST ACC (%)	VAL ACC$_{avg}$ (%)	RUN$_{time}$ (M)	RAM$_{avg}$ (MB)	RAM$_{max}$ (MB)	DISK$_{avg}$ (MB)	DISK$_{max}$ (MB)	CL_{score}
UT_LG	0.92	0.68	68.67	10643.25	11624.87	0	0	0.694
JODELET	0.88	0.64	6.59	15758.62	18169.32	0	0	0.680
AR1	0.80	0.58	20.46	8040.47	10092.72	0	0	0.663
YC14600	0.91	0.65	64.88	16425.64	19800.48	0	0	0.653
ICT_VIPL	0.95	0.68	76.73	2459.31	2459.68	392.187	562.5	0.617
SOONY	0.88	0.63	120.33	14533.97	15763.60	0	0	0.612
REHEARSAL	0.75	0.52	22.87	19056.77	23174.11	0	0	0.570
JIMIB	0.91	0.74	242.12	17995.61	23765.51	0	0	0.542
NOOBMASTER	0.76	0.53	147.59	24714.06	30266.62	0	0	0.464
NAÏVE	0.23	0.24	5.16	15763.46	18158.02	0	0	0.327
AVG	0.80	0.59	77.54	14539.12	17327.49	39.22	56.25	0.58

continual learning competition ever organized with a raised benchmark complexity and targeting real-word applications in computer vision. This challenge allowed every continual learning algorithm to be fairly evaluated with shared and unifying criteria and pushing the CL community to work on more realistic benchmarks than the more common MNIST or CIFAR.

After a carefully investigation and analysis of the competition results we can conclude that continual learning algorithms are mostly ready to face real-world settings involving high-dimensional video streams. This is mostly thanks to hybrid approaches often combined with plain replay mechanisms. However, it remains unclear if such techniques can scale over longer data sequences and without such an extensive use of replay.

Despite the significant participation and success of the 1st edition of the challenge, a number of possible improvements and suggestions for future continual learning competitions can be formulated:

- *Discourage over-engineered solutions*: one of the main goal of the competition was to evaluate the applicability of current continual learning algorithms on real-world computer vision problems. However, given the substantial freedom given through the competition rules to achieve this goal, we have noticed a number of over-engineered solutions aimed at improving the CL_{score} but not really significant in terms of novelty of scientific interest. This in turns forced every other participants to focus on over-engineering rather than on the core continual learning issues. For example, data loading or compression algorithms may be useful to decrease memory and compute overheads but may be applicable to most of the solutions proposed, making them less interesting and out of the scope of competition. For this reason, we believe that finding a good trade-off between realism and scientific interest of the competition will be fundamental for future challenges in this area. We suggest for example to block the possibility to optimize the data loading algorithms and to count the number of replay patterns rather than their

bytes overhead.

- *Automatize evaluation*: in the current settings of the challenge the evaluation was client-side (on the participants machines) for the pre-selection phase and on a server-side shared hardware for the finals. To ensure the fairness of the results and the competition rules adherence, the code that generated each submission had to be included as well. However, an always-available remote docker evaluation similar to the one proposed for the AnimalAI Olympics (Crosby et al., 2019), would allow a single phase competition with an always coherent and updated scoreboard, stimulating in turns teams participation and retention over the competition period. This would also alleviate some burdens at the organization levels, reducing the amount of manual interventions.

- *Add scalability metrics*: An interesting idea to tame the challenge complexity while still providing a good venue for assessing continual learning algorithms advancement, would be to include other than the already proposed metrics, a number of derivative ones taking into account their trend over time rather than their absolute value. This would help to better understand their scalability on more complex problems and longer tasks/batches sequences and incentivize efficient solutions with constant memory/computation overheads.

- *Encourage the focus on original learning strategies*: Another important possible improvement of the competition would be setting up a number of incentives and disincentives to explore interesting research directions in continual learning. For example, the usage of pre-trained models has been extensively used for the competition by all the participants. However it would have been also interesting to see proposals not taking advantage of it as well. In the next competition edition we plan to discourage the use of pre-trained models, different hyperparameters for each setting track and increase the memory usage weight associated to the CL_{score}.

Acknowledgements

References

Bae, H., Brophy, E., Chan, R. H., Chen, B., Feng, F., Graffieti, G., Goel, V., Hao, X., Han, H., Kanagarajah, S., et al. Iros 2019 lifelong robotic vision: Object recognition challenge [competitions]. *IEEE Robotics & Automation Magazine*, 27(2):11–16, 2020.

Buda, M., Maki, A., and Mazurowski, M. A. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

Caccia, L., Belilovsky, E., Caccia, M., and Pineau, J. Online learned continual compression with adaptive quantization modules. 2019.

Caccia, M., Rodriguez, P., Ostapenko, O., Normandin, F., Lin, M., Caccia, L., Laradji, I., Rish, I., Lacoste, A., Vazquez, D., et al. Online fast adaptation and knowledge accumulation: a new approach to continual learning. *arXiv preprint arXiv:2003.05856*, 2020.

Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P. K., Torr, P. H. S., and Ranzato, M. On tiny episodic memories in continual learning, 2019.

Chen, Y., Diethe, T., and Flach, P. Bypassing gradients re-projection with episodic memories in online continual learning. *arXiv preprint arXiv:2006.11234*, 2020.

Crosby, M., Beyret, B., and Halina, M. The animal-ai olympics. *Nature Machine Intelligence*, 1(5):257–257, 2019.

Díaz-Rodríguez, N., Lomonaco, V., Filliat, D., and Maltoni, D. Don't forget, there is more than forgetting: new metrics for Continual Learning. *Continual Learning Workshop at NIPS*, 2018. ISSN 00954918. doi: arXiv:1810.13166v1. URL http://arxiv.org/abs/1810.13166.

Díaz-Rodríguez, N., Lomonaco, V., Filliat, D., and Maltoni, D. Don't forget, there is more than forgetting: new metrics for continual learning. *arXiv preprint arXiv:1810.13166*, 2018.

Escalante, H. J., Tu, W.-W., Guyon, I., Silver, D. L., Viegas, E., Chen, Y., Dai, W., and Yang, Q. Automl@ neurips 2018 challenge: Design and results. In *The NeurIPS'18 Competition*, pp. 209–229. Springer, 2020.

Farquhar, S. and Gal, Y. Differentially private continual learning. *arXiv preprint arXiv:1902.06497*, 2019.

French, R. M. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Hayes, T. L., Kemker, R., Cahill, N. D., and Kanan, C. New metrics and experimental paradigms for continual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2031–2034, 2018.

Hayes, T. L., Cahill, N. D., and Kanan, C. Memory efficient experience replay for streaming learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9769–9776. IEEE, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lesort, T., Lomonaco, V., Stoian, A., Maltoni, D., Filliat, D., and Díaz-Rodríguez, N. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information Fusion*, 58:52–68, jun 2020. ISSN 15662535. doi: 10.1016/j.inffus.2019. 12.004. URL https://linkinghub.elsevier. com/retrieve/pii/S1566253519307377.

Lomonaco, V. *Continual Learning with Deep Architectures*. PhD thesis, alma, apr 2019. URL http:// amsdottorato.unibo.it/9073/.

Lomonaco, V. and Maltoni, D. CORe50: a New Dataset and Benchmark for Continuous Object Recognition. In *CoRL*, pp. 1–10, 2017a.

Lomonaco, V. and Maltoni, D. CORe50: a New Dataset and Benchmark for Continuous Object Recognition. In *Proceedings of the 1st Annual Conference on Robot Learning (CoRL)*, volume 78, pp. 17–26, 2017b. URL http://proceedings.mlr.press/ v78/lomonaco17a.html.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pp. 6467–6476, 2017.

Mai, Z., Kim, H., Jeong, J., and Sanner, S. Batch-level experience replay with review for continual learning. *arXiv preprint arXiv:2007.05683*, 2020.

Maltoni, D. and Lomonaco, V. Continuous learning in single-incremental-task scenarios. *Neural Networks*, 116: 56–73, 2019.

McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Mirzadeh, S. I., Farajtabar, M., Pascanu, R., and Ghasemzadeh, H. Understanding the role of training regimes in continual learning. *arXiv preprint arXiv:2006.06958*, 2020.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, may 2019. ISSN 08936080. doi: 10.1016/j.neunet.2019.01.012. URL https://doi.org/10.1016/j.neunet.2019. 01.012https://linkinghub.elsevier. com/retrieve/pii/S0893608019300231.

Pasquale, G., Ciliberto, C., Odone, F., Rosasco, L., and Natale, L. Are we done with object recognition? The iCub robot's perspective. *Robotics and Autonomous Systems*, 112:260–281, 2019. ISSN 09218890. doi: 10.1016/j.robot.2018.11.001.

Pellegrini, L., Graffieti, G., Lomonaco, V., and Maltoni, D. Latent Replay for Real-Time Continual Learning. *International Conference on Intelligent Robots and Systems (IROS)*, 2020.

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. Pascale 2 chal-lenge: Learning when test and training inputs have different distribu-tions challenge. 2005.

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. *Dataset shift in machine learning*. The MIT Press, 2009.

Richard, M. D. and Lippmann, R. P. Neural network classifiers estimate bayesian a posteriori probabilities. *Neural Computation*, 3(4):461–483, 1991.

Robins, A. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

Russakovsky, O., Deng, J., Su, H., Krause, J., and Satheesh, S. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Apr 2015. ISSN 1573-1405. doi: 10.1007/s11263-015-0816-y. URL http://dx.doi. org/10.1007/s11263-015-0816-y.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

She, Q., Feng, F., Hao, X., Yang, Q., Lan, C., Lomonaco, V., Shi, X., Wang, Z., Guo, Y., Zhang, Y., Qiao, F., and Chan, R. H. M. OpenLORIS-Object: A robotic vision dataset and benchmark for lifelong deep learning. In *2020 International Conference on Robotics and Automation (ICRA)*, pp. 4767–4773, 2020.

Yalniz, I. Z., Jégou, H., Chen, K., Paluri, M., and Maha-
jan, D. Billion-scale semi-supervised learning for image
classification. *arXiv preprint arXiv:1905.00546*, 2019.

A. Additional Details

In this appendix, additional details for each team and track are reported (see Tab. 6, Tab. 7 and Tab. 7). In particular we report: i) the model type; ii) if the model was pre-trained; iii) the type of strategy used; iv) the number of eventual replay examples; v) the number of training epochs per batch; vi) the mini-batch size used.

Table 6. Approaches and baselines details for the NI track.

Team	Model	Pre-trained	Strategy	Replay Examples$_{max}$	Epochs	Mini-batch size
UT_LG	DenseNet-161	yes	rehearsal	80000	2	32
Yc14600	ResNeSt50	yes	regularization & rehearsal	12000	1	16
ICT_VIPL	WideResNet-50	yes	rehearsal	4000	2	80
Jodelet	ResNet-50	yes	rehearsal	6400	1	32
Soony	ResNext101/50 & DenseNet161	yes	architectural & rehearsal	119894	1	900
JimiB	resnext101	yes	regularization & rehearsal	11989	8	32
Jun2tong	ResNet-50	yes	regularization & rehearsal	12000	5	32
Sahinyu	Efficientnet-b7	yes	rehearsal	8000	2	27
Ar1	mobilenetV1	yes	architectural	1500	4	128
Noobmaster	resnet-101	yes	rehearsal	24000	5	32
MrGranddy	EfficientNet-B7	yes	regularization & architectural	0	1	32
Naïve	mobilenetV1	yes	n.a.	0	4	128
Rehearsal	mobilenetV1	yes	rehearsal	160	4	128
HaoranZhu	ResNet-50	yes	regularization	0	10	32

Table 7. Approaches and baselines details the MT-NC track.

Team	Model	Pre-trained	Strategy	Replay Examples$_{max}$	Epochs	Mini-batch size
Ar1	mobilenetV1	yes	architectural & rehearsal	1500	4	128
UT_LG	DenseNet-161	yes	architectural	0	1	32
Yc14600	ResNeSt50	yes	regularization & rehearsal	4500	1	16
Soony	ResNext101/50 & DenseNet161	yes	architectural & rehearsal	119890	3	100
Jodelet	ResNet-50	yes	rehearsal	6400	1	32
jun2tong	ResNet-50	yes	regularization & rehearsal	45000	1	32
ICT_VIPL	ResNeXt-50	yes	rehearsal	3000	1	32
Rehearsal	mobilenetV1	yes	rehearsal	180	4	128
JimiB	resnext101	yes	regularization & rehearsal	11989	8	32
MrGranddy	EfficientNet-B7	yes	regularization & architectural	0	1	32
Noobmaster	resnet-101	yes	rehearsal	18000	5	32
HaoranZhu	ResNet-50	yes	regularization	0	10	32
Naïve	mobilenetV1	yes	n.a.	0	4	128

Table 8. Approaches and baselines details for the NIC track.

Team	Model	Pre-trained	Strategy	Replay Examples$_{max}$	Epochs	Mini-batch size
UT_LG	DenseNet-161	yes	rehearsal	78200	1	32
Jodelet	ResNet-50	yes	rehearsal	6400	1	32
Ar1	mobilenetV1	yes	architectural & rehearsal	1500	4	128
ICT_VIPL	ResNet50	yes	rehearsal	2000	1	64
Yc14600	ResNeSt50	yes	regularization & rehearsal	19550	1	32
Rehearsal	mobilenetV1	yes	rehearsal	7820	4	128
Soony	ResNext101/50 & DenseNet161	yes	architectural & rehearsal	119890	1	900
JimiB	resnext101	yes	regularization & rehearsal	11989	6	32
Noobmaster	resnet-101	yes	rehearsal	23460	5	32
Naïve	mobilenetV1	yes	n.a.	0	4	128