Supplementary material for
Ternary Feature Masks: zero-forgetting for task-incremental learning

Marc Masana
Computer Vision Center
Barcelona, Spain
mmasana@cvc.uab.es

Tinne Tuytelaars
ESAT-PSI
KU Leuven, Belgium
tinne.tuytelaars@esat.kuleuven.be

Joost van de Weijer
Computer Vision Center
Barcelona, Spain
joost@cvc.uab.es

A. More on related work

We show in Table S1 a comprehensive overview of some of the characteristics that we consider to be more relevant to the experimental setup we propose.

B. Ternary Mask Implementation

The formulation of our proposed method in Sec. 3 states that we can combine both masks m^t and n^t into a ternary mask. In order to make the implementation easier and more efficient, this is done by creating a ternary mask that is set to be state 2 for all features at task 1. This means that the first task will work as a normal network that allows for learning the task at hand as if we were using finetuning. Then, when moving to task 2, we will grow the network and add new features. The masks for task 1 associated with the new features will be set to state 0, therefore these are not used when evaluating nor are they learnable for task 1. The masks for task 2 will then be created by setting the previous existing features to state 1 and the new added features to state 2. This would allow the features with state 1 to be used during the forward pass for mask $n^{t=2}$, while the features with state 2 will contribute to both forward pass for mask $n^{t=2}$ and backward pass for mask $m^{t=2}$ (using Eq. 6). This process is explained in Algorithm S1.

Algorithm S1 : Growing Ternary Feature Masks

Input: ternary mask m for each task and layer
Input: s number of features to add per layer
Require: Network with L layers
Require: Tasks $1, \dots, T$, current task $t > 1$
% Loop for each layer %
for $l = 1, \dots, L$
  old_size $\leftarrow$ current_output_size(l)
  new_size $\leftarrow$ old_size + s^l
  % For each previous task %
  for $k = 1, \dots, t-1$
    $m^{k,l}$[old_size:new_size] $\leftarrow 0$
  end for
  % For the current task %
  $m^{t,l}$[0:old_size] $\leftarrow 1$
  $m^{t,l}$[old_size:new_size] $\leftarrow 2$
end for

C. Experimental Setup details

Datasets. Tiny ImageNet is a $64 \times 64 \times 3$ resized version of 200 ImageNet classes. ImageNet uses $256 \times 256 \times 3$ inputs that use random cropping to $224 \times 224 \times 3$ during training for data augmentation. In the case of Birds we resize the bounding box annotations of the objects to $224 \times 224 \times 3$ for all splits. We do the same for Actions but without using the bounding box annotations. For Flowers we resize to $256 \times 256 \times 3$ and also do data augmentation by random cropping $224 \times 224 \times 3$ patches during training and using the central crop for evaluation. In all experiments we perform random horizontal flips during training for data augmentation.

We decide to not do experiments on permuted MNIST since it has been shown to not allow fair comparison between different approaches [7]. The MNIST data contains a too large amount of zeros per input, which leads to an easy identification of important weights that can be frozen to not overlap with the other tasks. Furthermore, the MNIST data might be too simple to represent more realistic scenarios.

Hyperparameters. Distillation and model-based approaches use hyperparameters to control the trade-off between forgetting and intransigence on the knowledge of previous tasks. On top of that, LwF has a temperature scaling hyperparameter for the cross-entropy loss. From the mask-based models, HAT has a trade-off hyperparameter too and a maximum for the sigmoid gate steepness. PackNet has a prune percentage of the layers. We consider the values proposed by the papers if they have results on the same experimental setup. Otherwise, we use the hyperparameter search proposed in [3] and [13], which chooses hyperparameters with only the information at hand for each task.

Table S1. Summary of related work characteristics. *pf*: per feature, *pp*: per parameter, *pt*: per task, *pfp*: per feature and parameter.

Family	Method	Revisit data	Require Backbone net	Easily expandable	Overhead	Forgetting	Forward transfer	Features or weights
Baseline	Finetune	No	No	Yes	None	Yes	Little	neither
	Joint	Yes	No	Yes	None	No	Little	neither
	Freeze	No	Yes	Yes	None	No	Backbone only	neither
Distillation	LwF [8]	No	No	No	1 float *pp*	Some	Yes	weights
	LFL [4]	No	No	No	1 float *pp*	Some	Yes	weights
	PNN [16]	No	No	Yes	duplicate *pt*	Some	Yes	weights
	P&C [17]	No	No	No	extra network	Some	Little	weights
Model-based	EWC [5]	No	No	No	1 float *pp*	Some	Yes	weights
	R-EWC [9]	No	No	No	1 float *pp*	Some	Yes	weights
	IMM [7]	No	No	No	1 float *pp pt*	Some	Yes	weights
	SI [21]	No	No	No	1 float *pp*	Some	Yes	weights
	MAS [1]	No	No	No	1 float *pp*	Some	Yes	weights
	SSL [2]	No	No	No	1 float *pfp*	Some	Yes	both
Mask-based	PackNet [12]	No	No	No	1 int *pp pt*	No	Yes	weights
	PiggyBack [11]	No	Yes	No	1 bit *pp pt*	No	Backbone only	weights
	HAT [18]	No	No	Yes	1 float *pf pt*	Some	Yes	features
	TFM w/o FN (Ours)	No	No	Yes	2 bits *pf pt*	No	Yes	features
	TFM (Ours)	No	No	Yes	2 bits + 2 floats *pf pt*	No	Yes	features

D. A note on choosing expansion rates

The continual learning philosophy states the rule of not using data from previous tasks when learning new ones; only data of the current task can be used at each step of the setup. It is also common in machine learning setups to use a part of the training set as validation in order to choose the best hyper-parameters. Therefore, when learning each task, a validation set of that specific task at hand can be used to train the network avoiding overfitting, but no other data can be used (neither from test nor from other previous or future tasks). It is important to state that we strictly comply to these rules in the experiments we propose.

As explained in Section 3.4, our proposed approach can be expanded as needed in order to learn the new tasks without having to change the connections from previous tasks. However, this flexibility of choosing how many features will be added to each layer can easily become a rabbit-hole of architecture optimization. Because of that, we decide to propose a simple setup for how we apply our proposed approach to be comparable to the other state-of-the-art. We take the maximum layer size for all approaches to be the same as the VGGnet or AlexNet architectures for the experiments in Section 4. This way, all approaches will have a similar number of parameters.

The differences between Figures 2 and 3 are further explained in Fig. S1. During the first task, the only used features are those that have a grey-border in them. This means that for the two shown layers, task 1 is learned by using 8 features (with 12 connections). Once task 2 arrives, we fix the grey-border features and expand the network with the green-border ones. The new task then uses the existing

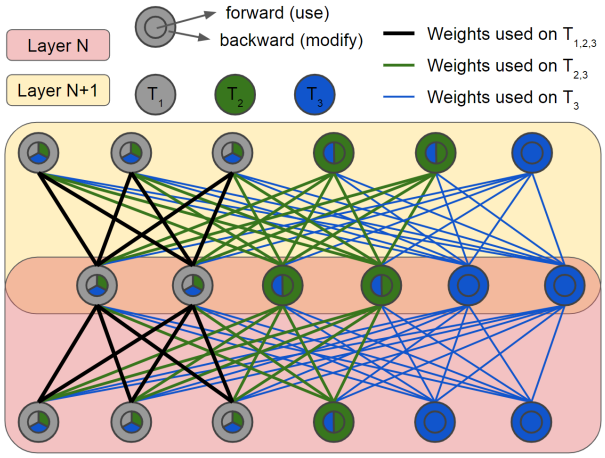

Figure S1. Network growth with ternary feature masks over three tasks.

network and expands it with 5 features (with 24 new connections). The masks for the green-border features are set to unused for masks of task 1, while they are learnable for task 2. Finally, as task 3 comes, 5 features are also added (with 36 new connections). Masks corresponding to the new features are set to unused for tasks 1 and 2, while set to learnable for task 3. This way of expanding the network also shows that as we learn more tasks, more knowledge is available from previous ones. This opens the possibility of having to add less and less features over time since the addition of each feature creates more connections to learn. In practice, one can imagine that when learning new tasks that are very similar to previous ones, no new features will have to be added and the current network knowledge and a spe-

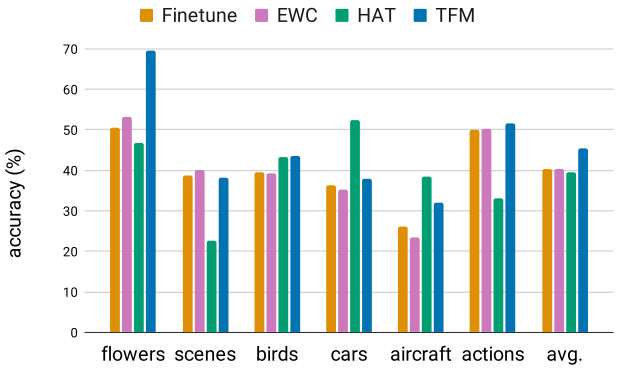

Figure S2. Comparison on a sequence of multiple datasets on AlexNet from scratch.

cific head for the task will be enough. Further research and analysis on the specific details for each layer expansion and architecture is left for future work.

E. Tiny ImageNet semantic splits

The semantically similar splits of tiny ImageNet used in Table 4 experiments are grouped as described in Table S2.

F. Extended results for Tiny ImageNet

We present in Tables S3, S4 and S5 more detailed accuracy and forgetting results for the experiment on Tiny ImageNet with random split, semantic split and larger first task, respectively. Both Tables S3 and S4 have a sequence of 10 tasks with 10 classes each, but with different class orderings. Table S5 shows results when starting with a larger first task, and thus the last column is only averaged over the smaller tasks (T2-T10). Results show that mask-based approaches achieve a better overall performance than other approaches on all splits, with TFM having the best performance or tied with the best.

G. Multi-dataset experiment

To further explore the performance of our proposed method when the tasks have a much different distribution, we shows results on a sequence of multiple fine-grained datasets: Oxford Flowers [14], MIT Indoor Scenes [15], CUB-200-2011 Birds [19], Stanford Cars [6], FGVC Aircraft [10], and Stanford Actions [20]. We compare TFM with the Finetuning baseline, EWC and HAT on AlexNet trained from scratch. Each of the approaches seems to have a tendency on doing better in different tasks. However, TFM is never the worse at any task and has a clear better average accuracy after learning all tasks. This seems to indicate that our proposed method can still perform quite well even when there are larger distribution changes between tasks.

Table S2. Semantically similar splits for tiny ImageNet.

Task	Semantic group	Classes
1	Animals (flying & insects)	scorpion, black widow, tarantula, spider web, centipede, trilobite, grasshopper, stick insect, cockroach, mantis, ladybug, dragonfly, monarch butterfly, sulphur butterfly, fly, bee, goose, black stork, king penguin, albatross.
2	Artifacts (smaller)	abacus, binoculars, candle, chain, chest, dumbbell, hourglass, lampshade, magnetic compass, pill bottle, computer keyboard, acorn, plunger, syringe, teddy bear, torch, comic book, remote control, umbrella, nail.
3	Music, Sport and Kitchen	basketball, punching bag, rugby ball, scoreboard, stopwatch, volleyball, CD player, drumstick, iPod, oboe, organ, refrigerator, cask, plate, wooden spoon, teapot, frying pan, beaker, bucket, dining table.
4	Animals (land)	brown bear, red panda, koala, pig, ox, bison, bighorn sheep, gazelle, dromedary, African elephant, orangutan, chimpanzee, baboon, cougar, lion, European fire salamander, bullfrog, tailed frog, American alligator, boa constrictor.
5	Artifacts (bigger)	altar, maypole, bannister, flagpole, fountain, parking meter, pay-phone, pole, cash machine, birdhouse, reel, bathtub, rocking chair, potter's wheel, sewing machine, space heater, turnstile, memorial tablet, desk, vestment.
6	Food	water jug, wok, guacamole, ice cream, lollipop, pretzel, mashed potato, cauliflower, bell pepper, mushroom, orange, lemon, banana, pomegranate, meat loaf, pizza, potpie, espresso, soda bottle, beer bottle.
7	Animals (water & pets)	dugong, goldfish, jellyfish, brain coral, American lobster, spiny lobster, sea slug, sea cucumber, guinea pig, snail, slug, poodle, Chihuahua, Yorkshire terrier, golden retriever, Labrador retriever, German shepherd, tabby cat, Persian cat, Egyptian cat.
8	Clothes and wearables	academic gown, poncho, apron, backpack, bikini, bow tie, fur coat, gasmask, kimono, sock, military uniform, miniskirt, neck brace, Christmas stocking, sombrero, sunglasses, cardigan, snorkel, sandal, swimming trunks.
9	Transport	bullet train, station wagon, freight car, go-kart, rickshaw, lifeboat, limousine, moving van, police van, school bus, convertible, crane, trolleybus, sports car, tractor, gondola, broom, cannon, lawn mower, missile.
10	Buildings and scenes	barbershop, barn, lighthouse, butcher shop, candy store, water tower, triumphal arch, suspension bridge, steel arch bridge, viaduct, thatched roof, cliff dwelling, dam, obelisk, picket fence, cliff, coral reef, lakeside, seacoast, alp.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[2] Rahaf Aljundi, Marcus Rohrbach, and Tinne Tuytelaars. Selfless sequential learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019. 2

[3] 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 (TPAMI)*, 2021. 1

[4] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim.

Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016. 2, 5

[5] 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. 2, 5

[6] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *International Conference on Computer Vision Workshops*, 2013. 3

[7] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in Neural Information Processing Systems*, 2017. 1, 2, 5

[8] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017. 2, 5

[9] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *International Conference on Pattern Recognition (ICPR)*, 2018. 2

[10] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013. 3

[11] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[12] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 5

[13] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation. *arXiv preprint arXiv:2010.15277*, 2020. 1

[14] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *IEEE Indian Conference on Computer Vision, Graphics & Image Processing*, 2008. 3

[15] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 3

[16] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 2

[17] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning (ICML)*, 2018. 2

[18] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning (ICML)*, 2018. 2, 5

[19] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 3

[20] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011. 3

[21] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning (ICML)*, 2017. 2

Table S3. Comparison with the state-of-the-art. Tiny ImageNet on VGGnet from scratch. Accuracy of each task after learning all tasks. Number between brackets indicates forgetting. Classes are randomly split and fixed for all approaches.

Tiny ImageNet - classes randomly split											
Approach	Task 1 (1-20)	Task 2 (21-40)	Task 3 (41-60)	Task 4 (61-80)	Task 5 81-100)	Task 6 (101-120)	Task 7 (121-140)	Task 8 (141-160)	Task 9 (161-180)	Task 10 (181-200)	Avg. all
Finetuning	38.1 (-13.6)	36.0 (-13.7)	43.2 (-16.0)	44.1 (-18.6)	45.5 (-12.6)	54.5 (-13.6)	50.3 (-15.7)	50.5 (-13.4)	51.0 (-13.1)	61.2 (0.0)	47.4
Freezing	51.7 (0.0)	36.4 (0.0)	39.5 (0.0)	41.7 (0.0)	42.9 (0.0)	46.2 (0.0)	45.7 (0.0)	41.1 (0.0)	41.2 (0.0)	40.9 (0.0)	42.7
Joint	58.6 (+6.9)	53.9 (+8.3)	59.1 (+3.7)	61.8 (+7.9)	57.7 (+2.9)	66.0 (+2.6)	64.0 (+3.1)	60.2 (+5.9)	57.9 (+1.0)	53.8 (0.0)	59.3
LfL [4]	32.4 (-18.9)	35.4 (-17.0)	43.4 (-15.7)	44.1 (-20.2)	45.0 (-15.0)	55.9 (-14.5)	49.4 (-16.1)	51.1 (-12.4)	58.6 (-8.0)	61.4 (0.0)	47.7
LwF [8]	45.1 (-6.6)	45.5 (-2.2)	53.5 (-4.6)	57.6 (-2.6)	56.2 (0.0)	65.7 (+0.4)	63.5 (-0.3)	58.4 (-1.9)	59.6 (-0.3)	58.5 (0.0)	56.4
IMM-mode [7]	50.6 (-1.1)	38.5 (+0.3)	44.7 (-0.1)	49.2 (+0.3)	47.5 (+1.1)	51.9 (-1.4)	53.7 (-0.6)	47.7 (-0.4)	50.0 (-2.2)	48.7 (0.0)	48.3
EWC [5]	33.9 (-17.4)	35.4 (-14.4)	43.6 (-15.4)	46.7 (-15.9)	49.5 (-9.1)	52.5 (-15.8)	47.8 (-20.0)	50.2 (-13.8)	56.6 (-9.9)	61.4 (0.0)	47.8
HAT [18]	46.8 (-0.2)	49.1 (+0.8)	55.8 (+0.2)	58.0 (-0.2)	53.7 (+0.3)	61.0 (+0.1)	58.7 (0.0)	54.0 (-0.1)	54.6 (-0.1)	50.3 (0.0)	54.2
PackNet [12]	52.5 (0.0)	49.7 (0.0)	56.5 (0.0)	59.8 (0.0)	55.0 (0.0)	64.7 (0.0)	61.7 (0.0)	55.9 (0.0)	55.2 (0.0)	52.5 (0.0)	**56.4**
TFM w/o FN (Ours)	49.6 (0.0)	47.2 (0.0)	54.8 (0.0)	58.2 (0.0)	55.0 (0.0)	64.0 (0.0)	59.3 (0.0)	53.6 (0.0)	55.5 (0.0)	51.9 (0.0)	54.9
TFM (Ours)	48.2 (0.0)	47.7 (0.0)	56.7 (0.0)	58.2 (0.0)	54.8 (0.0)	62.2 (0.0)	61.5 (0.0)	57.3 (0.0)	58.5 (0.0)	54.8 (0.0)	56.0

Table S4. Comparison with the state-of-the-art. Tiny ImageNet on VGGnet from scratch. Accuracy of each task after learning all tasks. Number between brackets indicates forgetting. Classes are split by semantic closeness and fixed for all approaches.

Tiny ImageNet - classes semantically split											
Approach	Task 1 fly anim.	Task 2 small artif.	Task 3 hobbies	Task 4 land anim.	Task 5 big artif.	Task 6 food	Task 7 pets/aquatic	Task 8 wearables	Task 9 transport	Task 10 scenes	Avg. all
Finetuning	17.1 (-34.2)	19.7 (-17.7)	20.9 (-24.5)	16.7 (-30.5)	20.8 (-28.7)	29.2 (-22.0)	30.7 (-21.0)	25.2 (-17.8)	40.2 (-18.9)	59.9 (0.0)	28.0
Freezing	51.3 (0.0)	28.5 (0.0)	27.2 (0.0)	29.6 (0.0)	29.0 (0.0)	35.0 (0.0)	31.7 (0.0)	23.9 (0.0)	37.0 (0.0)	34.7 (0.0)	32.8
Joint	55.0 (+3.7)	41.9 (+7.9)	46.2 (+6.3)	44.9 (+4.9)	44.7 (+7.1)	49.0 (+4.8)	46.6 (+4.9)	36.4 (+4.7)	51.2 (+5.7)	51.1 (0.0)	46.7
LfL [4]	17.2 (-34.1)	18.4 (-21.0)	21.5 (-24.0)	18.7 (-30.5)	20.2 (-28.6)	27.4 (-23.9)	28.4 (-22.3)	26.0 (-18.4)	41.2 (-17.6)	59.1 (0.0)	27.8
LwF [8]	34.0 (-13.9)	18.4 (-14.5)	32.6 (-0.8)	36.5 (-5.6)	40.1 (-0.5)	43.1 (-2.5)	41.8 (-1.3)	32.7 (-1.1)	50.3 (-0.5)	48.1 (0.0)	37.8
IMM-mode [7]	42.3 (-9.0)	28.8 (+0.1)	26.5 (-3.1)	30.7 (-3.6)	32.5 (-3.1)	28.8 (-13.0)	35.4 (-6.2)	27.3 (-3.7)	43.6 (-4.9)	42.7 (0.0)	33.9
EWC [5]	20.2 (-31.1)	18.5 (-19.3)	20.2 (-26.3)	20.9 (-28.9)	24.7 (-22.8)	25.5 (-27.5)	28.7 (-23.4)	23.0 (-19.6)	39.8 (-20.2)	56.8 (0.0)	27.8
HAT [18]	44.6 (+0.4)	34.8 (+0.2)	40.8 (-0.1)	45.4 (+0.4)	40.8 (-2.5)	49.8 (0.0)	44.9 (-0.2)	33.1 (-1.7)	51.9 (0.0)	53.8 (0.0)	44.0
PackNet [12]	47.0 (0.0)	35.7 (0.0)	42.7 (0.0)	48.6 (0.0)	45.8 (0.0)	48.1 (0.0)	45.9 (0.0)	38.3 (0.0)	51.2 (0.0)	49.1 (0.0)	45.2
TFM w/o FN (Ours)	46.4 (0.0)	34.7 (0.0)	38.8 (0.0)	44.1 (0.0)	42.0 (0.0)	48.3 (0.0)	46.5 (0.0)	35.7 (0.0)	52.0 (0.0)	54.8 (0.0)	44.3
TFM (Ours)	46.4 (0.0)	37.2 (0.0)	40.4 (0.0)	44.1 (0.0)	44.2 (0.0)	48.2 (0.0)	46.4 (0.0)	37.5 (0.0)	53.5 (0.0)	54.7 (0.0)	**45.3**

Table S5. Comparison with the state-of-the-art. Tiny ImageNet on VGGnet from scratch. Accuracy of each task after learning all tasks. Numbers between brackets indicates forgetting. LArger first task. Average on the smaller tasks 2 to 10.

Tiny ImageNet - larger first task											
Approach	Task 1 (1-110)	Task 2 (111-120)	Task 3 (121-130)	Task 4 (131-140)	Task 5 (141-150)	Task 6 (151-160)	Task 7 (161-170)	Task 8 (171-180)	Task 9 (181-190)	Task 10 (191-200)	Avg. (111+)
Finetuning	18.4 (-33.2)	39.6 (-34.4)	56.6 (-21.0)	58.0 (-23.8)	44.6 (-32.8)	63.0 (-21.8)	51.0 (-27.2)	51.4 (-19.0)	70.4 (-14.0)	80.0 (0.0)	57.2
Freezing	51.6 (0.0)	68.6 (0.0)	70.2 (0.0)	77.9 (0.0)	68.1 (0.0)	78.8 (0.0)	72.9 (0.0)	64.2 (0.0)	76.7 (0.0)	70.2 (0.0)	69.9
LfL [4]	16.7 (-33.9)	58.3 (-6.6)	59.5 (-2.9)	64.6 (-2.2)	58.2 (-1.4)	64.7 (-0.6)	63.4 (+0.5)	54.4 (+0.1)	61.0 (0.0)	56.5 (0.0)	60.0
LwF [8]	10.8 (-40.0)	29.5 (-43.5)	44.6 (-30.4)	61.2 (-21.2)	55.5 (-15.1)	73.3 (-8.8)	71.7 (-3.5)	62.3 (-1.8)	77.8 (-2.2)	74.3 (0.0)	61.1
IMM-mode [7]	26.1 (-26.3)	50.4 (-13.8)	59.5 (-19.3)	61.6 (22.8)	54.0 (-21.8)	63.2 (-22.5)	58.2 (-19.9)	56.0 (-13.9)	75.0 (-6.7)	79.9 (0.0)	62.0
EWC [5]	51.8 (-0.7)	24.8 (-0.1)	43.3 (-1.2)	61.8 (-0.5)	55.5 (-0.3)	70.8 (-0.7)	67.6 (-0.1)	53.8 (-0.6)	70.1 (-1.2)	61.7 (0.0)	56.6
HAT [18]	46.1 (0.0)	60.1 (-0.1)	68.2 (+0.2)	73.2 (+0.1)	63.2 (+0.1)	76.2 (+0.1)	67.4 (0.0)	58.1 (-0.1)	73.2 (0.0)	59.1 (0.0)	66.5
PackNet [12]	47.6 (0.0)	74.0 (0.0)	74.2 (0.0)	79.0 (0.0)	65.2 (0.0)	76.2 (0.0)	69.4 (0.0)	61.4 (0.0)	73.4 (0.0)	64.8 (0.0)	70.8
TFM w/o FN (Ours)	49.6 (0.0)	69.8 (0.0)	71.1 (0.0)	79.8 (0.0)	68.4 (0.0)	78.4 (0.0)	72.9 (0.0)	64.8 (0.0)	75.9 (0.0)	70.2 (0.0)	72.4
TFM (Ours)	49.9 (0.0)	70.4 (0.0)	71.4 (0.0)	80.8 (0.0)	70.5 (0.0)	79.4 (0.0)	73.9 (0.0)	64.4 (0.0)	76.5 (0.0)	72.1 (0.0)	**73.3**