Supplementary Material:
Tangent Model Composition for Ensembling and Continual Fine-tuning

A. Implementation Details

We run all our experiments on a ResNet-50 model pre-trained with ImageNet, except in Table. 3 where we use a pre-trained ResNet-18 model instead for fair comparison with existing benchmarks. To construct the tangent models, we re-initialize the last fully connected layer such that the number of output classes matches that of the target tasks. We train the original models with SGD using the cross-entropy loss. Tangent models are trained with the Adam optimizer using the rescaled squared loss (RSL). We train all (original and tangent) models for 50 epochs using a decaying learning rate factor of 0.1 at epochs 25 and 40, and use a constant batch size of 32 across all experiments. We average each experiment across 3 independent runs.

For all baseline methods, we follow [3, 4, 2] and standardize batch size at 32. The only two exceptions are Co2L [5], where we set batch size and buffer batch size to 256 since a large batch size is needed to achieve reasonable accuracy, and TWF [2] where we set both task and buffer batch size to 12 due to its larger memory requirement resulting from the attention mechanism. For replay-based baselines, at each iteration, in addition to sampling a batch from the current task, we also sample a batch of the same size (32) from the replay buffer. Note that this immediately increases training time for all replay-based methods by a factor of two, since the training set size for each task is effectively doubled for fixed number of epochs.

We prepare our datasets as follows: **(a) Caltech-256 [6]** contains $\sim 30.6K$ images across 256 object classes. Following [1], we sample 60 images per class to train and test on the remaining. **(b) MIT-67 [8]** contains $\sim 15.6K$ images across 67 indoor scene categories. We split MIT-67 using the provided train-test splits. **(c) OxfordPets [7]** contains $\sim 7.3K$ images across 37 pet categories. We split the dataset equally for training and testing.

We set $\lambda_{(t,1)} = \frac{1}{t}$ and $\lambda_{(t,2)} = \frac{t-1}{t}$, which weights each model by the number of component models used to construct it. This is a natural choice when tasks are assumed to be distributed uniformly. We note that in cases such as task imbalance, it is likely that there will be more effective choices for λ. We leave the investigation of this for future work.

B. Extended Literature Review

We describe in detail the three scenarios for continual learning proposed by [9]. In Task-Incremental Learning, task identity (Task-ID) is provided at inference time. In other words, models can be trained with task-specific components which are "selected" during inference by the provided Task-ID. For example, in the case of multi-class classification, this corresponds to simply selecting the output nodes corresponding to the task, and restricting predictions only to this subset. Thus, preserving intra-task performance is critical for Task-Incremental Learning. In Data/Domain-Incremental Learning, Task-ID is not provided at inference time. However, knowledge of the task identity is not necessary for inference. [9] cites the example of protocols under which structure of each task remains consistent, but distribution of inputs differs across tasks. Lastly, Class-Incremental Learning requires both inferring Task-ID and solving the task at hand. For instance, this happens when each task contains a new subset of classes/labels that are not present in previously encountered tasks.

C. Additional Discussion

C.1. Fine-tuning classification head

The effectiveness of TMC is partly due to the implicit regularization arising from linearization of a model about the pre-training objective. In the extreme case, we can choose only to linearize with respect to the last layer of a neural network, which in our scenario is the classification head. Note that this is often a fully-connected layer which is already linear. As such, this is equivalent to simply fine-tuning the classification head, which can be composed for inference. We refer to this as

Table 1. Non-linear fine-tuning on the DomainNet dataset.

Test Domain	Train Domain						Composition		
	Clipart	Quickdraw	Photo	Infograph	Sketch	Painting	Ens-L	Ens-SM	Soup
Clipart	75.90± 0.10	4.56± 1.42	48.06± 0.27	38.32± 0.26	49.60± 0.68	42.57± 0.91	60.27± 6.84	70.68± 0.13	62.18± 0.42
Quickdraw	9.45± 0.25	69.64± 0.08	4.85± 0.13	3.35± 0.23	10.56± 0.16	3.73± 0.12	25.11± 0.48	36.43± 0.47	14.30± 0.21
Photo	54.21± 0.32	1.68± 0.38	83.41± 0.08	53.27± 0.45	48.93± 0.41	60.20± 0.49	62.19± 11.88	77.51± 0.23	69.97± 0.36
Infograph	18.47± 0.24	0.55± 0.17	21.84± 0.11	43.25± 0.24	15.33± 0.16	19.34± 0.52	24.05± 3.96	33.13± 0.26	26.04± 0.14
Sketch	41.40± 0.77	5.08± 0.52	37.00± 0.20	29.88± 0.36	69.14± 0.16	37.25± 0.28	54.54± 2.86	61.37± 0.22	52.88± 0.42
Painting	36.83± 0.45	0.70± 0.11	48.48± 0.16	35.70± 0.34	36.70± 0.52	71.53± 0.33	46.14± 12.21	63.64± 0.54	55.14± 0.64
Average	39.38± 0.35	13.70± 0.45	40.61± 0.16	33.96± 0.31	38.37± 0.35	39.10± 0.44	45.39± 6.37	57.13± 0.31	46.75± 0.36

Table 2. Tangent Fine-tuning on the DomainNet dataset.

Test Domain	Train Domain						Composition	
	Clipart	Quickdraw	Photo	Infograph	Sketch	Painting	TME	TMC
Clipart	64.01± 0.15	3.52± 0.09	34.15± 0.17	24.43± 0.23	33.07± 0.17	27.70± 0.09	50.54± 0.17	57.10± 0.23
Quickdraw	2.57± 0.10	56.70± 0.07	1.71± 0.06	1.15± 0.03	2.97± 0.07	1.19± 0.07	7.17± 0.15	25.54± 0.63
Photo	45.68± 0.13	2.71± 0.02	80.31± 0.06	47.08± 0.04	41.49± 0.10	54.18± 0.06	69.34± 0.06	74.50± 0.05
Infograph	12.94± 0.08	0.75± 0.04	15.37± 0.15	33.66± 0.10	10.43± 0.08	12.89± 0.09	21.52± 0.13	23.63± 0.15
Sketch	25.24± 0.12	3.08± 0.06	23.10± 0.02	17.41± 0.19	54.65± 0.13	22.11± 0.05	37.43± 0.11	44.38± 0.09
Painting	26.62± 0.05	1.68± 0.01	40.74± 0.13	27.97± 0.12	28.73± 0.20	63.26± 0.06	48.99± 0.09	55.36± 0.03
Average	29.51± 0.10	11.41± 0.05	32.56± 0.10	25.29± 0.12	28.56± 0.12	30.22± 0.07	39.17± 0.12	46.75± 0.20

TMC-FC, and show in Table 3 that this simplest form of linearization can yield even better accuracies than TMC when the pre-training features are sufficient for the downstream task (in the case of C-Caltech-256), but performs significantly worse when this assumption does not hold (C-MIT-67 and C-OxfordPets).

Table 3. **Fine-tuning classification head:** Fine-tuning only the classification head and composing them is the simplest form of TMC (we refer to this as TMC-FC), where linearization is only done with respect to the last fully-connected classification layer. Here, we show that TMC-FC is highly effective for C-Caltech-256 using an ImageNet pre-training objective, but TMC works significantly better for C-MIT-67 and C-OxfordPets since the features learnt from ImageNet pre-training do not generalize as well to indoor scene recognition and fine-grained object (pet species) classification respectively.

Dataset	Tasks	Soup	Ens-L	Ens-SM	TMC-FC	TMC	TME
Inference/Memory Cost:		$\mathcal{O}(1)$	$\mathcal{O}(T)$	$\mathcal{O}(T)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(T)$
C-Caltech-256 *(Class-IL)*	5	84.00	82.80	83.09	**85.30**	84.82	*85.53*
	10	80.10	76.85	79.76	**83.43**	82.37	*82.58*
	20	74.86	66.76	75.47	**79.13**	78.66	*79.30*
C-MIT-67 *(Class-IL)*	5	59.48	56.79	58.68	61.50	**69.43**	*71.72*
	10	49.30	42.36	50.52	54.53	**64.53**	*65.72*
	20	28.51	23.61	35.62	40.05	**48.98**	*54.93*
C-OxfordPets *(Class-IL)*	5	80.71	81.14	81.86	77.83	**84.42**	*85.04*
	10	71.35	69.71	77.31	70.93	**77.75**	*79.97*

C.2. Failure modes of TMC

As discussed in the main paper, while TMC and TME is often more effective than weight averaging or ensembling of non-linear models partly due to the implicit regularities provided by model linearization, this can lead to over-regularization when the downstream task and pre-training objective are highly dissimilar. We demonstrate this using the DomainNet dataset consisting of 6 different domains - Cliparts, Google Quickdraws, Photos, Infographs, Sketches, and Paintings. We fine-tune, from an ImageNet pre-training initialization, individual non-linear and tangent models on each domain in Table 1 and Table 2 respectively. We show that in such cases, component models and the composed model trained using non-linear fine-tuning outperforms tangent fine-tuning, as a result of over-regularization in the tangent model.

C.3. Visualizations of Component Model Accuracies

We provide additional visualizations of the individual component model accuracies for MIT-67, Caltech-256, and Oxford-Pets in Fig. 1, Fig. 2, and Fig. 3 respectively.

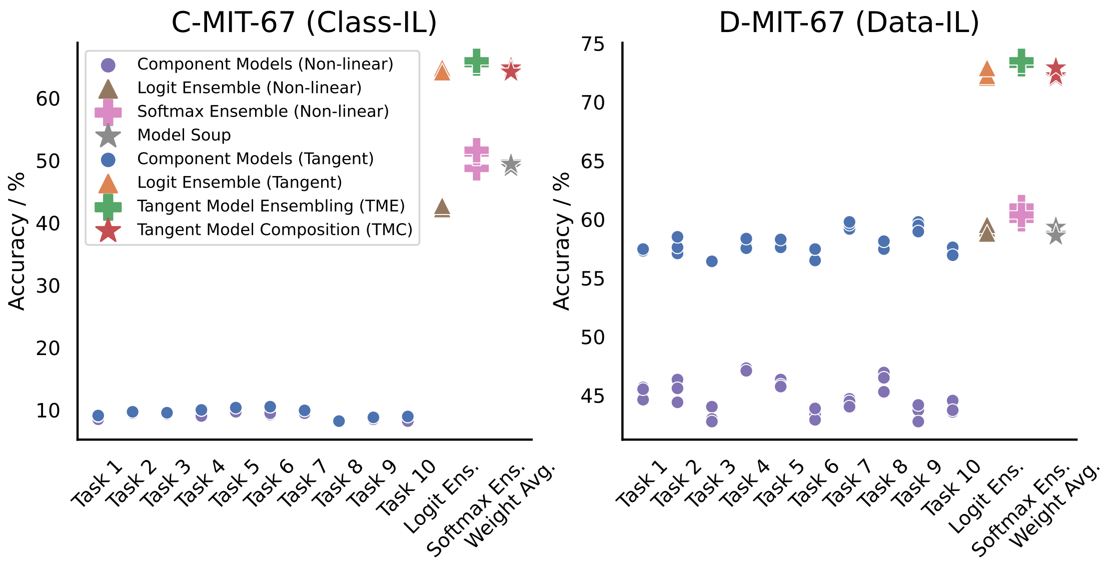

Figure 1. Accuracies of component and composed models for MIT-67 Dataset

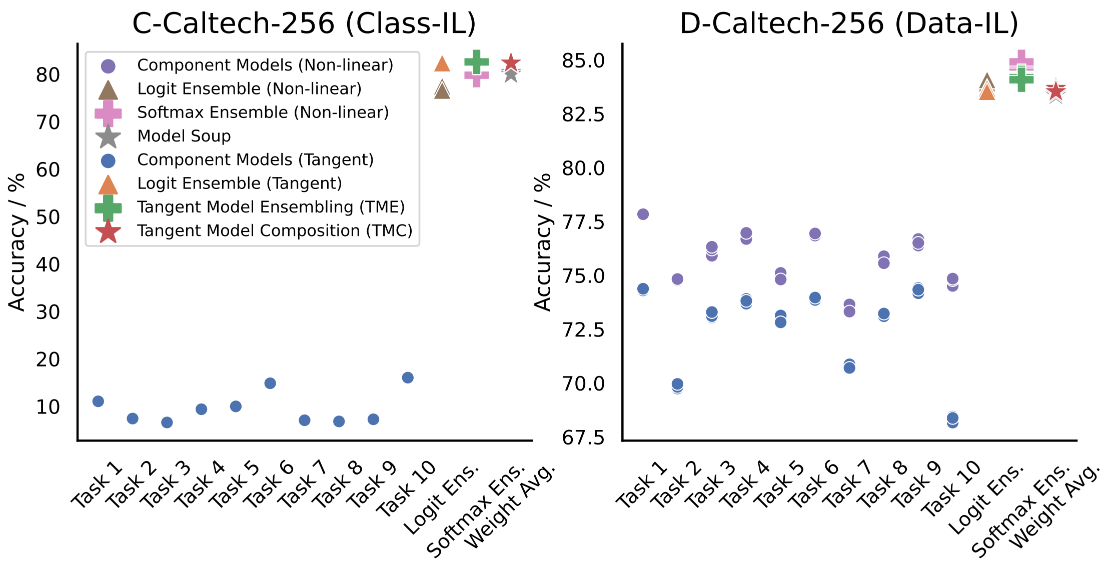

Figure 2. Accuracies of component and composed models for Caltech-256 Dataset

C.4. Further Discussion on the Rescaled Square Loss

Here, we elaborate on the effectiveness of RSL when tasks are highly dissimilar (class-incremental learning). First, note that due to random initialization, component models trained on tasks $\tau \neq t$ will produce non-zero noisy output signals for labels contained within task t upon composition. While these noisy signals are too small to harm or bias the predictions of

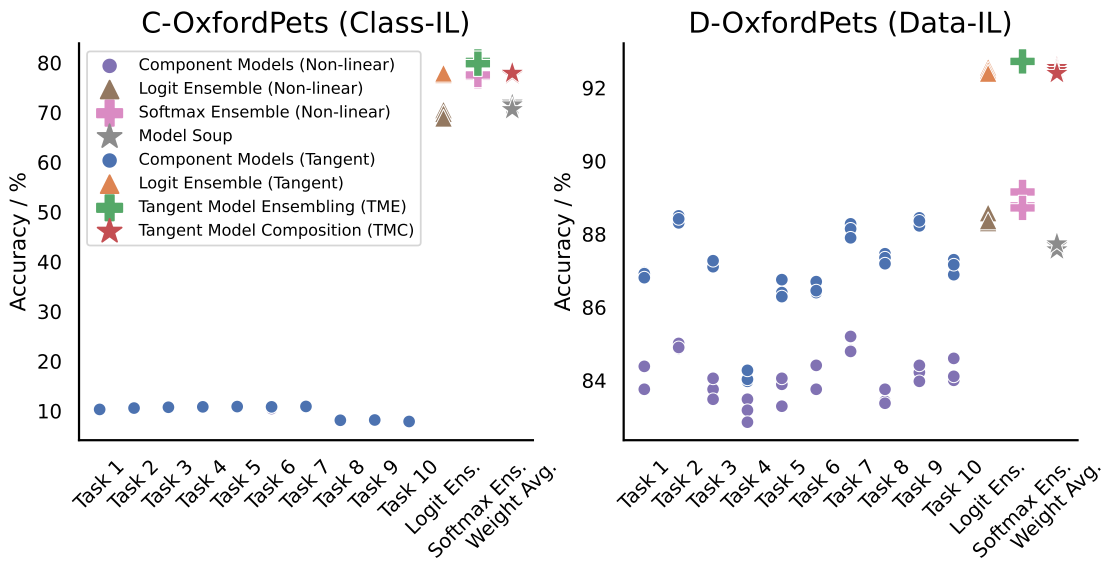

Figure 3. Accuracies of component and composed models for OxfordPets Dataset

individual component models, the summation of the noise from each component model upon ensembling/composition can significantly affect the final prediction.

The effectiveness of soft-max ensembling demonstrated in the main paper can be attributed to the fact that the soft-max operation greatly increases the signal-to-noise ratio for each individual component model. This minimizes the sum of the noise components relative to the constructive signals present in the final composed model. Training using RSL with large values of β aims to mimic this effect by encouraging greater separation between the positive and negative outputs, and thus increasing the signal-to-noise ratio of each component model.

We show the effects of β on the output distribution of the composed model in Fig. 4. Lower values of β reduces the signal-to-noise ratio, causing noisy signals to become more prominent in the final composed model. On the other hand, larger values of β increases the distinction between the positive and negative signals in the final composed model.

On the other hand, in the case when tasks are similar, values of β can result in ignoring weak but constructive signals from different component models.

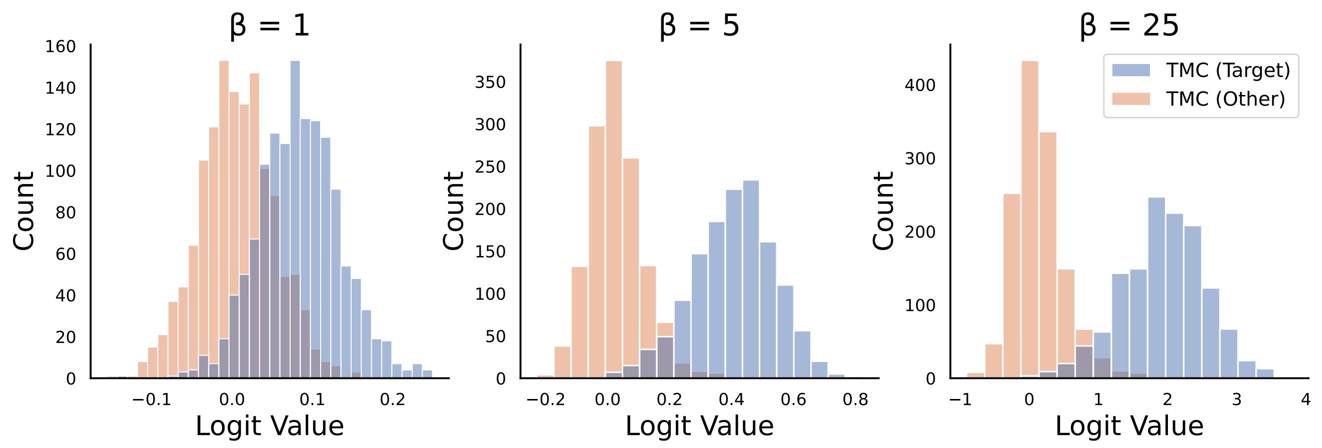

Figure 4. We plot the output logit distribution pertaining to the ground-truth class **[TMC (Target)]** and another class **[TMC (Other]**. Output values for the former should ideally be large, while output values for the latter should ideally be close to zero. We see that for small $\beta = 1$, there is significant overlap between the two distributions, reducing the contrast between positive and negative output signals. On the other hand, larger values of β produces significantly less overlap between the two distributions. Plots are done on C-MIT-67, 10 tasks.

References

[1] Alessandro Achille, Aditya Golatkar, Avinash Ravichandran, Marzia Polito, and Stefano Soatto. Lqf: Linear quadratic fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15729–15739, 2021. 1

[2] Matteo Boschini, Lorenzo Bonicelli, Angelo Porrello, Giovanni Bellitto, Matteo Pennisi, Simone Palazzo, Concetto Spampinato, and Simone Calderara. Transfer without forgetting. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIII*, pages 692–709. Springer, 2022. 1

[3] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020. 1

[4] Pietro Buzzega, Matteo Boschini, Angelo Porrello, and Simone Calderara. Rethinking experience replay: a bag of tricks for continual learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2180–2187. IEEE, 2021. 1

[5] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 9516–9525, 2021. 1

[6] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007. 1

[7] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012. 1

[8] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE conference on computer vision and pattern recognition*, pages 413–420. IEEE, 2009. 1

[9] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019. 1