Functional Task Networks: Cortex-Inspired Spatial Parameter Isolation for Continual Learning

Anonymous Authors[1]

Abstract

Continual adaptation is hardest under concept shift, where successive tasks share an input distribution but require incompatible outputs. We introduce Functional Task Networks (FTN), a parameter-isolation method that routes each task through a sparse subset of independent components without requiring a task ID at inference. Given a small labeled support batch, FTN constructs a binary mask by optimizing mask logits, applying spatial smoothing on a component grid, and selecting a fixed-capacity set of active components. The same cold-start procedure is used to allocate components during training and to recover them at evaluation, so task inference is handled by the adaptation mechanism itself. Disjoint masks yield disjoint gradient paths, giving a structural no-forgetting guarantee under standard optimizer precautions. We instantiate the same idea in LoRA by replacing a single low-rank adapter with a mask-routed pool of rank-r components. On synthetic concept-shift benchmarks, FTN substantially reduces forgetting relative to shared-parameter baselines while preserving task performance. On a six-task distil-gpt2 style-transformation stream, LoRA-FTN reduces forgetting from +7.37 to +0.07 nats under support-batch mask recovery, approaching an oracle disjoint-adapter reference without using task labels.

1. Introduction

Modern learning systems, from lifelong agents to continually pretrained language models (Gururangan et al., 2020; Yıldız et al., 2024), must absorb new data without overwriting old competence. When a network trained on task A is fine-tuned on task B, gradient updates can overwrite the configuration that encoded A (McCloskey & Cohen, 1989; French, 1999), producing *catastrophic forgetting*. The problem is sharpest under *block-sequential* training, in which each task is a contiguous data block (the standard regime for continual pretraining of LLMs and for instruction- or preference-tuning (Ouyang et al., 2022; Rafailov et al., 2023)), and under *concept shift*, in which the same input features map to different correct outputs across tasks (Gama et al., 2014). The learner must additionally solve a task-inference problem: given a few examples, identify which prior solution applies, without an oracle task label (Aljundi et al., 2019; van de Ven & Tolias, 2019).

Three families dominate the continual-learning literature (Parisi et al., 2019): *regularization* approaches (Kirkpatrick et al., 2017; Zenke et al., 2017), which penalize updates to important parameters but degrade under concept shift; *experience replay* (Rebuffi et al., 2017; Lopez-Paz & Ranzato, 2017), which scales in memory with task history; and *parameter isolation* (Mallya & Lazebnik, 2018; Serrà et al., 2018; Wortsman et al., 2020; Rusu et al., 2016; Yoon et al., 2018), which assigns each task a disjoint subnetwork. Many parameter-isolation methods require an oracle task identity at inference or learn a separate task-embedding-conditioned router across tasks; SupSup is a close exception, because it can infer a task by optimizing over learned supermasks when task identity is absent (Wortsman et al., 2020). We introduce *Functional Task Networks* (FTN), whose single short inner-loop optimization both installs new subnetworks during training and re-selects prior ones at inference. No discrete task label and no learned cross-task router are needed; recovery does require a small labeled support batch from the current task to drive the gradient proposal.

Contributions. (1) A distributed mixture-of-experts type architecture whose disjoint masks give a structural no-forgetting guarantee (Theorem 2.1). (2) A three-stage mask configurer (gradient proposal, lateral smoothing, k-WTA) that is run cold-started at both training and inference. (3) Synthetic experiments showing FTN-Slow reaches approximately zero forgetting under the realistic mask-recovery protocol; FixedMask reaches exactly zero with

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the ICML 2026 Workshop "Continual Adaptation at Scale: Towards Sustainable AI". Do not distribute.

the optimizer-state reset of Section 3. (4) The same mask machinery, dropped into LoRA as *LoRA-FTN* (Section 2), gives an LLM continual-fine-tuning method that empirically realises Theorem 2.1 on `distilgpt2` across six concept-shift style tasks: $\sim 100\times$ less forgetting than a shared LoRA.

2. Method

Model. We place H "neurons" on a square cortical grid of side $D = \sqrt{H}$. Each neuron $k \in \{1, \dots, H\}$ is itself a private feed-forward network with L hidden layers of inner width d_{inner} and a scalar output. Weights are stored as $H \times d_{\text{in}} \times d_{\text{inner}}$ tensors and applied with a batched einsum that broadcasts the shared input x across neuron slots. A binary routing mask $m \in \{0,1\}^H$ gates the per-neuron scalar outputs; a single linear readout produces the prediction:

$$z_k = \text{MLP}_k(x), \qquad \hat{y} = W_{\text{out}}\,(\text{Dropout}(z) \odot m).$$

The model is a sparse mixture of H experts (Shazeer et al., 2017); weights and biases are not shared across neurons except through the readout columns.

Theorem 2.1 (Structural forgetting guarantee)**.** *Let $m^{(s)}, m^{(t)} \in \{0,1\}^H$ be the masks active during blocks $s < t$. If $m^{(s)} \odot m^{(t)} = \mathbf{0}$ then for every parameter θ_k of neuron k with $m_k^{(s)} = 1$ and every column $[W_{\text{out}}]_{:,k}$ the gradient of the task-t loss is zero. Training on block t leaves the task-s solution exactly unchanged.*

Proof. With $m_k^{(t)} = 0$, z_k is multiplied by zero in the loss, so $\partial \mathcal{L}_t / \partial z_k = 0$; by the chain rule the gradient is zero throughout θ_k and at $[W_{\text{out}}]_{:,k}$. □ □

Remark 2.2 (Implementation caveats). The theorem assumes that no parameter changes occur on inactive neurons via channels other than the explicit gradient: in particular, decoupled weight decay (AdamW) (Loshchilov & Hutter, 2019) and Adam first/second moments (Kingma & Ba, 2015) carried across task boundaries do change inactive parameters and produce a small $\text{FM} > 0$ even for FixedMask. Resetting the Adam state at every task boundary suffices; see Section 3.

Three-stage mask configurer. The mask is produced by Algorithm 1, given the live model and a single batch: (i) *Gradient proposal*: S Adam (Kingma & Ba, 2015) steps on the sigmoid-relaxed mask logits, cold-started from $m = \mathbf{0}$, minimize the task loss on the batch. (ii) *Lateral smoothing*: the proposal is reshaped to $D \times D$ and convolved T times with a uniform $s \times s$ kernel (torus padding); this projects the proposal onto the manifold of contiguous spatial blobs. (iii) *KWTA*: top-k binarisation gives a fixed capacity budget k/H per task. The same procedure is used to install a subnetwork during training and to recover one at inference; no task ID, task embedding, or learned cross-task router is required, but the configurer does need a small labeled support batch from the current task.

Algorithm 1 SMOOTHKWTA mask configurer

Input: model f_θ, batch (X, Y), side D, kernel s, lateral steps T, winners k, reconfig steps S, LR η
$m \leftarrow \mathbf{0}$
for $i = 1, \dots, S$ **do**
  $m \leftarrow m - \eta\, \widehat{\nabla}_m \mathcal{L}\big(f_\theta(X, \sigma(m)), Y\big)$
end for
Reshape m to $D \times D$; $m \leftarrow K_s^{(T)} * m$
return $\text{KWTA}(\text{flatten}(m),\, k)$

Variants. FTN-Fast uses a 17×17 kernel for T=2 steps (rapid spatial convergence); FTN-Slow uses a 3×3 kernel for T=15 steps (fine-grained iterative smoothing). KWTA-only is the ablation with T=0. FixedMask uses predetermined disjoint blocks of size k per task and is the structural upper bound. NoMask is the all-ones mask (naive fine-tuning). EWC (Kirkpatrick et al., 2017) is the regularization baseline (λ=400). *SupSup* (Wortsman et al., 2020) freezes the network at random init and learns one binary supermask per task via top-k straight-through; we use a labeled support batch (matching FTN's information access) and pick the best stored supermask by argmin loss at recovery. *SupSup-Wide* is the same baseline but with the signed Kaiming-He init from the SupSup paper and twice the per-neuron MLP width, the regime in which random supernets are typically more expressive.

The architecture is loosely cortex-inspired (per-neuron MLPs, horizontal smoothing, lateral inhibition, basal-ganglia-like gating); see Appendix A.

Adapting FTN to LoRA: LoRA-FTN. The structural ingredient of FTN, a binary mask gating an additive contribution from a pool of independent components, transfers directly to the LoRA parameterisation (Hu et al., 2022) of a frozen pretrained model. Replace the standard rank-r update $\Delta W = BA$ at each adapted weight with a structured pool of H rank-r components $\{(B_k, A_k)\}_{k=1}^H$ routed by the same binary mask $m \in \{0,1\}^H$:

$$\Delta W(m) \;=\; \sum_{k=1}^{H} m_k\, B_k A_k, \qquad y \;=\; (W + \Delta W(m))\, x. \tag{1}$$

The frozen base W is shared; A and B are the only trained parameters; m selects which components contribute. Standard LoRA initialisation (A uniform, B zero) makes ΔW initially zero. A single mask is shared across all adapted layers, so a "component" spans the network's depth, the LoRA

analogue of a parallel-neuron slot. Theorem 2.1 carries over verbatim: with $m_k^{(t)} = 0$ the term $m_k^{(t)} B_k A_k x$ is multiplied by zero in every forward pass, so $\partial \mathcal{L}_t / \partial A_k = \partial \mathcal{L}_t / \partial B_k = 0$. The same SmoothKWTA configurer installs and recovers the mask without modification. Full module details and the LoRA-restated theorem are in Appendix D.

3. Experiments

Setup. The model in experiment 1 is structured like a large mixture of experts with H=1024 slots on a 32×32 grid, L=8-layer per-neuron MLPs of inner width d_{inner}=8, output dropout 0.2, KWTA budget k=128 (12.5% of slots per task). Optimizer: Adam (Kingma & Ba, 2015) at 3×10^{-4} with weight decay disabled and the (m, v) moments reset at every task boundary. The reset is required for Theorem 2.1 to hold empirically; without it, carried-over Adam moments emit non-zero updates on inactive neurons even when their gradients are zero, producing a small but reproducible FM > 0 for FixedMask. With the reset, FixedMask attains FM $= 0.000 \pm 0.000$ exactly across all eight seeds (Tables 1 and 2). Eight seeds, single H100 GPU; mean $\pm$ std reported. The LoRA-FTN setup re-uses the same grid, sparsity, and configurer; model, hyperparameters, and tasks are summarised in Section 3 and Appendix E. Headline metrics use the *mask-recovery* protocol (the configurer is cold-started on a fresh evaluation batch with no task identity).

Experiment 1 (synthetic). A 2-D input is projected through a fixed encoder to a 24-D latent; each of three tasks selects a disjoint 8-D block via a hidden mask, and a sinusoidal nonlinearity at frequency 8 produces classification or regression targets. One epoch, 1000 steps per task, batch 256; the mask is reconfigured every training batch. Variants share the pipeline; they differ in readout/loss (CE vs. MSE) and configurer inner optimization (S=1, η_m=1.0 for clf; S=10, η_m=0.2 for reg). Tables 1 and 2 report mask-recovery metrics. FixedMask achieves FM $= 0$ on both variants per Theorem 2.1; FTN-Slow has the smallest FM among adaptive methods; SupSup attains FM $= 0$ but is capped by its frozen random base (ACC 0.645 / MSE 0.206, well below FTN-Slow); SupSup-Wide bumps clf only to 0.660 and *worsens* regression (reconfig FM $= 0.101 \pm 0.123$): regression mask recovery is intrinsically harder than classification recovery because per-task loss surfaces are less discriminable, and SupSup has no mechanism to adapt to this; FTN compensates by increasing S (S=10 for reg vs. S=1 for clf). Training the body matters: frozen-base methods pay an absolute-performance tax that FTN avoids while preserving the structural guarantee.

Where the loss goes. A single ACC/MSE conflates two failure modes: training-time overlap (subsequent training on shared neurons) and recall error (failing to re-find the trained mask at inference). Decomposing the prior-task loss against the FixedMask oracle on Exp. 1 (Appendix B, Table 4): NoMask/EWC pay everything to overlap; KWTA-only pays both terms (+0.13 ACC, +0.27 MSE recall error); only the FTN variants keep both terms small simultaneously. FTN-Slow has the smallest training-time overlap (slightly better than the FixedMask oracle on prior tasks at this seed budget) and FTN-Fast has the smallest recall error; the two trade off as a bias–variance pair.

Table 1. Experiment 1 (synthetic), classification variant; mean $\pm$ std over 8 seeds, mask-recovery protocol.

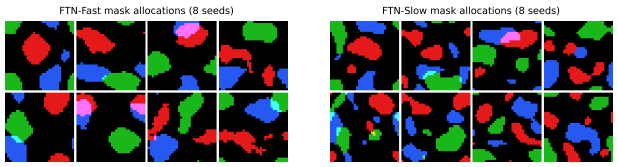
METHOD	ACC↑	FM↓
NOMASK	0.666 ± 0.031	0.466 ± 0.046
FIXEDMASK	0.938 ± 0.014	0.000 ± 0.000
KWTA	0.640 ± 0.062	0.152 ± 0.052
FTN-FAST	0.899 ± 0.037	0.042 ± 0.045
FTN-SLOW	0.896 ± 0.025	0.014 ± 0.025
EWC	0.665 ± 0.031	0.463 ± 0.046
SUPSUP	0.645 ± 0.028	0.000 ± 0.000
SUPSUP-WIDE	0.660 ± 0.020	0.000 ± 0.000

Table 2. Experiment 1 (synthetic), regression variant; mean $\pm$ std over 8 seeds, mask-recovery protocol. MSE: lower is better; FM is the average peak-MSE minus final-MSE on prior tasks.

METHOD	MSE↓	FM↓
NOMASK	0.312 ± 0.050	0.054 ± 0.071
FIXEDMASK	0.006 ± 0.002	0.000 ± 0.000
KWTA	0.589 ± 0.316	0.028 ± 0.038
FTN-FAST	0.126 ± 0.057	0.013 ± 0.017
FTN-SLOW	0.167 ± 0.075	0.003 ± 0.007
EWC	0.311 ± 0.050	0.054 ± 0.071
SUPSUP	0.206 ± 0.029	0.000 ± 0.000
SUPSUP-WIDE	0.216 ± 0.027	0.101 ± 0.123

Experiment 2 (continual LLM fine-tuning with LoRA-FTN). The intent of this experiment is twofold: (i) to test whether the structural forgetting guarantee of Theorem 2.1 carries over to a LoRA-parameterised pretrained model, and (ii) to test whether the gradient $\rightarrow$ smoothing $\rightarrow$ KWTA configurer can install and recover task-specific adapters cold-started on a small support batch when the underlying loss is a real autoregressive LM loss.

We continually, block-sequentially fine-tune a frozen `distilgpt2` (Radford et al., 2019; Sanh et al., 2019; Wolf et al., 2020; Hugging Face, 2019) (six transformer layers) on a stream of N=6 style-transformation tasks. Every block's attention QKV projection (`c_attn`) is wrapped with a LoRA-FTN pool of H=576 components on a 24×24 grid, rank r=4, KWTA winners k=72 (12.5% active per task). A single mask is shared across all wrapped layers. Optimizer: Adam (Kingma & Ba, 2015) at 5×10^{-4}, weight



(a) FTN-Fast, 8 seeds. *(b)* FTN-Slow, 8 seeds.

Figure 1. RGB mask allocations on synthetic classification. Each tile is one seed, each color one of the three tasks. The lateral kernel groups neurons into compact, approximately non-overlapping blobs that are topographically consistent across initialisations.

decay 0, (m, v) reset at every task boundary (Remark 2.2, restated for AdamW in Appendix D). Each task gets 200 training steps at batch 8; the configurer's support batch is 16 examples; we report per-token CE on a held-out test split, mean $\pm$ std over 7 random seeds. Headline metrics use the realistic mask-recovery protocol.

The six tasks share the same source corpus of $\sim$ 120 short mixed-case English sentences (a mix of pangrams and English proverbs, embedded in the experiment's source code so the run requires no external data download). Each task applies a different deterministic text-to-text transformation: `upper`, `lower`, `reverse_words`, `double_words`, `starred_vowels` (vowels replaced with $\star$), and `pirate` (`"Arr! ...matey!"` wrapping). Same input distribution, different target text $\Rightarrow$ strict concept shift. Four methods are compared: **LoRA-FTN** (the proposed method); **LoRA-FixedMask** (deterministic disjoint-block per task; structural oracle); **LoRA-Naive** (all-ones mask; a single shared LoRA of effective rank Hr); and **LoRA-Bank**, the simplest version of the FTN idea with no mask machinery, a pool of H=8 rank-288 LoRAs (matched per-task and total parameter capacity to LoRA-FTN), B initialised to small random Gaussian to break symmetry, and the configurer reduced to argmin of the support-batch loss over the H one-hot masks.

Caveat (LoRA initialisation). Standard LoRA init (B=0) makes untrained components contribute zero, so early in training the gradient proposal selects new-task components partly by *avoiding* harmful prior components rather than by positively identifying useful new ones; non-zero B init is a natural follow-up.

Table 3 summarises the result. LoRA-Naive collapses ($+7.37 \pm 0.50$ nats); LoRA-FixedMask attains FM $= 0.00$ exactly under both protocols, the empirical confirmation of Theorem 2.1 on the LoRA-parameterised model. LoRA-FTN retains within 0.07 ± 0.03 nats of the oracle under the realistic protocol, a $\sim 100\times$ reduction relative to LoRA-Naive; residual forgetting is concentrated on the gradient-similar pair (`reverse_words` / `double_words`) whose recovered masks share roughly a quarter of their winners.

Table 3. Block-sequential continual fine-tuning of `distilgpt2` on six mutually-exclusive style-transformation tasks (concept shift). Per-token cross-entropy on a held-out test split, mean $\pm$ std over 7 random seeds; FM is the average rise in CE on prior tasks above the lowest level they reached during training (positive $\Rightarrow$ forgetting). STORED uses each task's training-time mask (oracle); RECONFIG cold-starts the SmoothKWTA configurer on a fresh support batch with no task identity.

METHOD	PROTOCOL	CE↓	FM↓
LORA-NAIVE	STORED	8.22 ± 0.42	$+7.37 \pm 0.50$
	RECONFIG	8.22 ± 0.42	$+7.37 \pm 0.50$
LORA-FIXED	STORED	1.39 ± 0.09	0.00 ± 0.00
	RECONFIG	1.39 ± 0.09	0.00 ± 0.00
LORA-BANK	STORED	1.49 ± 0.22	$+0.11 \pm 0.30$
	RECONFIG	1.42 ± 0.09	$+0.04 \pm 0.11$
LORA-FTN	STORED	1.71 ± 0.16	$+0.34 \pm 0.11$
	RECONFIG	1.51 ± 0.08	$+0.07 \pm 0.03$

LoRA-Bank attains comparable mean retention (FM $= +0.04 \pm 0.11$ reconfig) but with markedly higher variance: $6/7$ seeds hit FM $= 0$ exactly (each task picks a unique slot), while $1/7$ seeds suffers a slot collision that drives FM to $+0.80$ stored / $+0.28$ reconfig, the failure mode the smoothing/KWTA machinery is designed to prevent. We read LoRA-FTN's value as enabling *distributed* task representations that are free to share components rather than enforcing the strict disjointness LoRA-Bank's argmin imposes; the present results show this distributed regime is competitive with strict disjointness, and we leave the question of transfer to held-out tasks to future work. Per-task matrices, forgetting curves, mask-overlap tables, and mask-tiling visualisations are in Appendix F.

4. Discussion

Discussion. Unlike PackNet, HAT, Progressive Networks, and DEN (Mallya & Lazebnik, 2018; Serrà et al., 2018; Rusu et al., 2016; Yoon et al., 2018), FTN needs no oracle task identity and no learned cross-task router; it is closer to SupSup (Wortsman et al., 2020) but trains the body and re-uses a single cold-start mask-search at both training and inference. Lateral smoothing biases KWTA subset-selection from $\binom{H}{k}$ toward $\mathcal{O}(H)$ compact regions, buying recall stability against gradient-proposal noise. Limitations: modest benchmarks; structural guarantee contingent on careful implementation (Remark 2.2, Appendix C); capacity bounded by H/k. Natural next steps: larger foundation models, more module types, conflicting-preference RLHF / instruction streams.

Impact Statement

This work contributes to continual learning, an essential feature of generally intelligent and adaptive systems and of localised AI that learns from streaming data.

References

Aljundi, R., Kelchtermans, K., and Tuytelaars, T. Task-free continual learning. *IEEE CVPR*, 2019.

Amari, S.-i. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27 (2):77–87, 1977.

Beniaguev, D., Segev, I., and London, M. Single cortical neurons as deep artificial neural networks. *Neuron*, 109 (17):2727–2739, 2021.

Douglas, R. J. and Martin, K. A. Neuronal circuits of the neocortex. *Annual Review of Neuroscience*, 27:419–451, 2004.

Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

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

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):44:1–44:37, 2014.

Gilbert, C. D. and Wiesel, T. N. Clustered intrinsic connections in cat visual cortex. *Journal of Neuroscience*, 3(5): 1116–1133, 1983.

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., and Smith, N. A. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Hubel, D. H. and Wiesel, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160(1):106–154, 1962.

Hugging Face. DistilGPT2 model card. https://huggingface.co/distilbert/distilgpt2, 2019. Accessed 2026-04-30.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Maass, W. On the computational power of winner-take-all. *Neural computation*, 12(11):2519–2535, 2000.

Mallya, A. and Lazebnik, S. Packnet: Adding multiple tasks to a single network by iterative pruning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

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

Mink, J. W. The basal ganglia: focused selection and inhibition of competing motor programs. *Progress in Neurobiology*, 50(4):381–425, 1996.

O'Reilly, R. C. and Frank, M. J. Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. *Neural Computation*, 18(2):283–328, 2006.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022.

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, 2019.

Poirazi, P. and Papoutsi, A. Illuminating dendritic function with computational models. *Nature Reviews Neuroscience*, 21(6):303–321, 2020.

Poirazi, P., Brannon, T., and Mel, B. W. Pyramidal neuron as two-layer neural network. *Neuron*, 37(6):989–999, 2003.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. *OpenAI technical report*, 2019.

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. icarl: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Rolls, E. T. Attractor networks. *Wiley Interdisciplinary Reviews: Cognitive Science*, 1(1):119–134, 2010.

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.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Schultz, W., Dayan, P., and Montague, P. R. A neural substrate of prediction and reward. *Science*, 275(5306): 1593–1599, 1997.

Serrà, J., Suris, D., Miron, M., and Karatzoglou, A. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pp. 4548–4557. PMLR, 2018.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *ICLR*, 2017.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

van de Ven, G. M. and Tolias, A. S. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45. Association for Computational Linguistics, 2020.

Wortsman, M., Ramanujan, V., Liu, R., Kembhavi, A., Rastegari, M., Yosinski, J., and Farhadi, A. Supermasks in superposition. *Advances in Neural Information Processing Systems*, 33:15173–15184, 2020.

Yıldız, Ç., Ravichandran, N. K., Sharma, N., Bethge, M., and Ermis, B. Investigating continual pretraining in large language models: Insights and implications. *arXiv preprint arXiv:2402.17400*, 2024.

Yoon, J., Yang, E., Lee, J., and Hwang, S. J. Lifelong learning with dynamically expandable networks. *ICLR*, 2018.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

A. Extended Biological Motivation

The architectural choices in Section 2 compress several cortical motifs that admit longer treatment.

Pyramidal neurons as deep MLPs. Cortical pyramidal neurons exhibit nonlinear dendritic processing comparable in expressivity to a small multi-layer artificial network (Poirazi et al., 2003; Poirazi & Papoutsi, 2020; Beniaguev et al., 2021). Modeling each "neuron" in our population as an L-layer MLP with a scalar output (an analogue of the cortical firing rate) compartmentalizes nonlinear feature extraction within neurons and reduces task-specific selection to a search over linear combinations. The scalar output bottleneck also discourages redundant solutions, mitigating the load-imbalance and expert-collapse problems familiar from mixture-of-experts (Shazeer et al., 2017; Fedus et al., 2022). Output dropout (Srivastava et al., 2014) is an algorithmic analogue of cortical response variability and grants tolerance to imperfect mask recovery.

Lateral excitation as smoothing. Cortical pyramidal neurons interact through dense local recurrent and horizontal connectivity (Hubel & Wiesel, 1962; Gilbert & Wiesel, 1983; Douglas & Martin, 2004); we abstract this as a uniform positive-valued convolution applied to the routing mask rather than to neural activations. This causes neurons to group by *task relevance* rather than by input similarity. Two consequences: spatial separation of task subnetworks (reducing overlap and interference) and reduction of the mask search space from $\binom{H}{k}$ toward $\mathcal{O}(H)$ under the single-blob idealization.

Lateral inhibition as KWTA. Cortical inhibitory interneurons enforce competition among excitatory cells (Maass, 2000; Douglas & Martin, 2004; Amari, 1977); we abstract this as a k-winner-take-all stage that produces a fixed capacity budget per subtask. A continuous variant (e.g., a Mexican-hat kernel of $\pm$-valued lateral connectivity) would soften k but is left to future work.

Gradient proposal as basal-ganglia gating. The initial gradient-based proposal of which neurons drive the loss plays the role of a reward-prediction-error / dopaminergic gating signal selecting among cortical assemblies (Mink, 1996; Schultz et al., 1997; O'Reilly & Frank, 2006; Rolls, 2010).

B. Mask-overlap vs. Recall-error Decomposition

The decomposition reported in Section 3 computes, for each method M and each prior task j ($j < N-1$),

$$\text{overlap}_M(j) = p^{\text{stored}}_{\text{FixedMask}}(j) - p^{\text{stored}}_M(j),$$
$$\text{recall}_M(j) = p^{\text{stored}}_M(j) - p^{\text{recfg}}_M(j),$$

on classification (signs reversed for regression MSE excess), then averages across prior tasks and seeds.

Table 4. Decomposition of Exp. 1 mask-recovery loss into training-time mask overlap and inference-time recall error against the FixedMask oracle. Entries are unitless drops (clf ACC) or excesses (reg MSE). FixedMask reference: clf ACC$=0.939$, reg MSE$=0.006$.

	Clf ACC drop		Reg MSE excess	
Method	Overlap	Recall	Overlap	Recall
NoMask	$+0.430$	$+0.000$	$+0.461$	$+0.000$
EWC	$+0.428$	$+0.000$	$+0.459$	$+0.000$
KWTA	$+0.154$	$+0.131$	$+0.272$	$+0.267$
FTN-Fast	$+0.016$	$+0.010$	$+0.098$	$+0.032$
FTN-Slow	-0.004	$+0.021$	$+0.110$	$+0.022$

C. Implementation Notes for Theorem 2.1

The theorem assumes that no parameter changes occur on inactive neurons. In practice, a number of common implementation choices break this assumption silently and produce a small but reproducible $\text{FM} > 0$ even for FixedMask:

- **Decoupled weight decay** (e.g. AdamW (Loshchilov & Hutter, 2019)): applies $\theta \leftarrow (1 - \eta\lambda)\theta$ regardless of gradient. Set weight decay to 0 or apply it only to active neurons.

- **Adam moment carry-over** (Kingma & Ba, 2015): at a task boundary the prior task's (m, v) moments are non-zero, so subsequent Adam updates on inactive neurons remain non-zero (decaying as $(\beta_1/\sqrt{\beta_2})^t$) for $\sim 1/(1-\beta_1/\sqrt{\beta_2}) \approx 10$ steps. Reset the optimizer state at every task boundary.

- **Shared output biases**: a W_{out} with bias couples all task readouts; we use bias-free readout. BatchNorm/statistics updates have the same problem.

With these precautions, FixedMask attains $\text{FM} = 0.000 \pm 0.000$ exactly across all 8 seeds on Exp. 1, and LoRA-FixedMask attains $\text{FM} = 0$ exactly on the LLM stream (Appendix D).

D. LoRA-FTN: Module, Theorem, and Implementation Notes

This appendix expands the brief LoRA adaptation in Section 2 into the full module, restates the structural forgetting theorem in the LoRA setting, and lists the implementation precautions needed for the theorem to hold empirically.

LoRA-FTN module, restated. For each adapted weight $W \in \mathbb{R}^{d_o \times d_i}$ in the frozen pretrained model, we replace the standard rank-r LoRA update $\Delta W = BA$ by a structured pool of H rank-r components $\{(B_k, A_k)\}_{k=1}^{H}$ with $A_k \in \mathbb{R}^{r \times d_i}$, $B_k \in \mathbb{R}^{d_o \times r}$. A single binary mask $m \in \{0,1\}^H$ with $\|m\|_0 = k$ routes the contributions:

$$\Delta W(m) \;=\; \sum_{j=1}^{H} m_j \, B_j A_j, \qquad y \;=\; (W + \Delta W(m))\, x. \tag{2}$$

The frozen base W is shared across tasks; A and B are the only trained parameters; m selects which components contribute on the current pass. Standard LoRA initialisation (A uniform, B zero) makes the initial ΔW identically zero. A single mask of size H is shared across all adapted layers, so a "component" j spans the full depth of the network, the LoRA analogue of a parallel-neuron slot in the basic mixture-of-experts type model.

Theorem D.1 (Structural forgetting, LoRA-FTN)**.** *Let $m^{(s)}, m^{(t)} \in \{0,1\}^H$ be the masks active during fine-tuning blocks $s < t$. If $m^{(s)} \odot m^{(t)} = \mathbf{0}$, then for every (A_j, B_j) with $m_j^{(s)} = 1$, the gradient of the block-t loss with respect to A_j and B_j is exactly zero. Training on block t leaves the block-s LoRA adapter unchanged.*

Proof. With $m_j^{(t)} = 0$, the term $m_j^{(t)} B_j A_j x$ is multiplied by zero in every forward pass through every adapted weight W, so $\partial \mathcal{L}_t / \partial A_j = 0$ and $\partial \mathcal{L}_t / \partial B_j = 0$. □ □

Implementation precautions for LoRA-FTN. The same precautions of Appendix C apply, with one substitution: many LLM fine-tuning pipelines default to AdamW (Loshchilov & Hutter, 2019) with non-zero weight decay, which silently violates the theorem on inactive components. We disable decoupled weight decay and reset the Adam (m, v) moments at every task boundary. With both precautions, LoRA-FixedMask attains $\text{FM} = 0.000$ exactly on every run of Section 3.

E. LoRA-FTN: Hyperparameters and Tasks

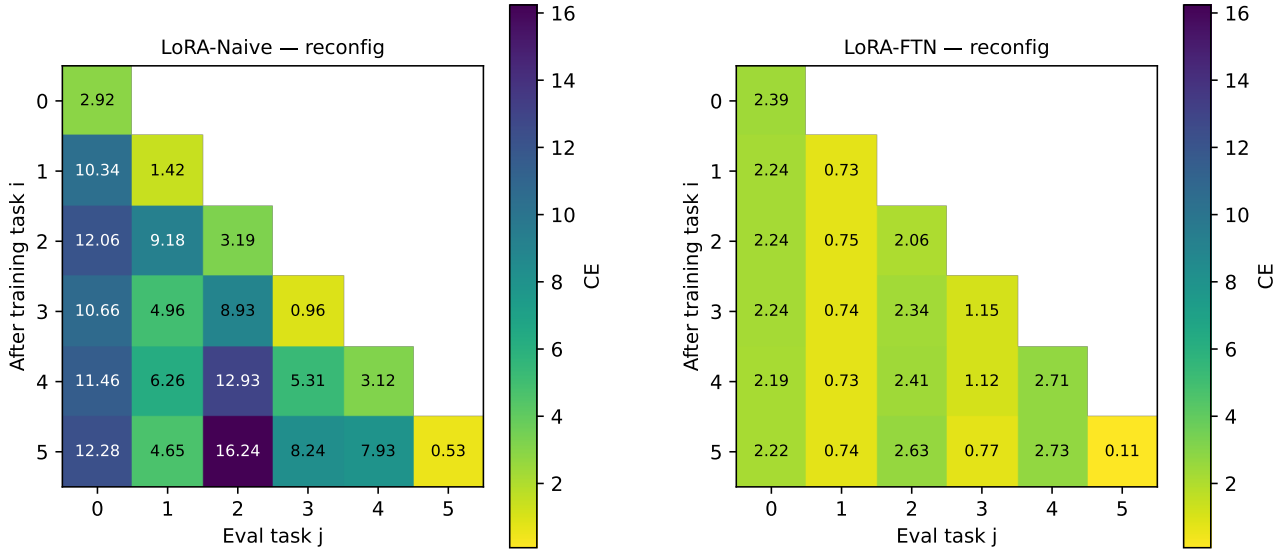
`distilgpt2` transformer (Radford et al., 2019; Sanh et al., 2019; Wolf et al., 2020; Hugging Face, 2019) (frozen, six layers); each block's attention QKV projection (`c_attn`) is wrapped with a LoRA-FTN pool. Pool size $H = 576$, grid side $D = 24$, rank $r = 4$, KWTA winners $k = 72$ (12.5% active per task). Optimizer Adam (Kingma & Ba, 2015) at $\eta = 5 \times 10^{-4}$, 200 training steps per task, batch 8, support-batch size 16, max sequence length 96. Adam (m, v) are reset at every task boundary; decoupled weight decay is 0 (Appendix D). SmoothKWTA configurer parameters: $S = 15$ inner steps at $\eta_m = 0.5$, kernel 3×3, $T = 15$ lateral steps. Table 3 reports mean $\pm$ std over 7 random seeds. The diagnostic mask-layout visualisations in Appendix F use eight random seeds. The full code is in `experiments/lora/`.

Tasks. The six tasks share the same source corpus of mixed-case English sentences and apply different deterministic text-to-text transformations: `upper`, `lower`, `reverse_words` (reverse word order), `double_words` (each word printed twice), `starred_vowels` (vowels replaced with $\star$), and `pirate` (`"Arr!  ...matey!"` wrapping). The input token distribution is identical across tasks; only the target text changes, giving strict concept shift.

F. LoRA-FTN: Per-task Diagnostics

This appendix contains the per-task forgetting pattern, per-task forgetting curves, the pairwise mask-overlap matrix, and the mask-tiling visualisation that complement the headline numbers in Table 3.

Per-task performance matrices. Figure 2 shows the per-task performance matrices under the realistic mask-recovery protocol. LoRA-Naive's matrix darkens monotonically along each prior-task column as more tasks are added, every cell off the diagonal is overwritten. LoRA-FTN's matrix retains the diagonal and most off-diagonals at low CE, with concentrated forgetting on a single pair (`reverse_words` / `double_words`), exactly the pair flagged by the mask-overlap diagnostic below.



(a) LoRA-Naive (RECONFIG). *(b)* LoRA-FTN (RECONFIG).

Figure 2. Per-task performance matrices on the LLM concept-shift stream (per-token CE on test, lower is better). Cell (i, j) is the loss on task j after training through task i. LoRA-Naive (left) overwrites every prior task; LoRA-FTN (right) retains nearly all prior tasks via mask recovery, with concentrated forgetting on a single gradient-similar pair.

Per-task forgetting curves. Figure 3 traces the per-task CE through the training stream: LoRA-Naive's curves climb sharply on every prior task whenever a new task is trained; LoRA-FixedMask's curves are flat (Theorem D.1); LoRA-FTN's curves stay flat for most pairs, with a single visible spike when a gradient-similar later task overwrites a shared component.

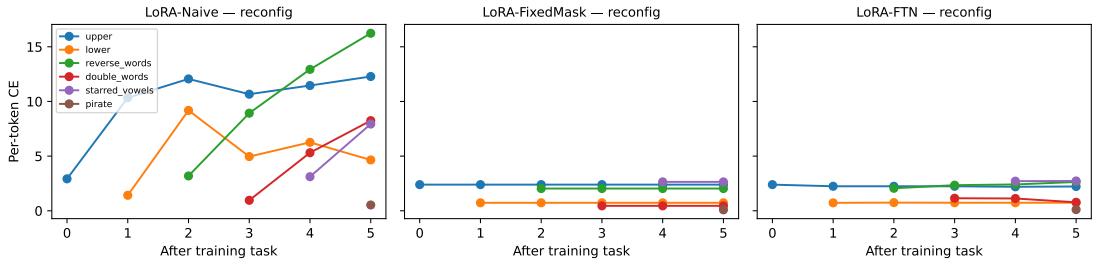


Figure 3. Per-task CE through the training stream (RECONFIG protocol). Each curve is one task; the x-axis is "after training through task i." The single visible spike on LoRA-FTN corresponds to the gradient-similar pair identified in Table 5.

Mask-overlap diagnostic. The disjoint-mask theorem makes a precise prediction: for any two tasks whose recovered masks are disjoint, training the second leaves the first untouched. Table 5 reports the pairwise mask overlap $|m_i \cap m_j|/k$ for the six LoRA-FTN training-time masks on the 24×24 grid. Most pairs are exactly disjoint. The largest overlap is 0.36

Table 5. Pairwise overlap of LoRA-FTN training-time masks: $|m_i \cap m_j|/k$, where $k = 72$ is the per-task active count. Diagonal is identically 1 and omitted; off-diagonal 0 means exact disjointness (Theorem 2.1 gives FM= 0 for that pair).

	upper	lower	reverse	double_	starred	pirate
upper	·	0.36	0.00	0.00	0.00	0.00
lower	0.36	·	0.00	0.00	0.00	0.06
reverse_wor	0.00	0.00	·	0.24	0.00	0.00
double_word	0.00	0.00	0.24	·	0.00	0.00
starred_vow	0.00	0.00	0.00	0.00	·	0.18
pirate	0.00	0.06	0.00	0.00	0.18	·

(`upper`/`lower`, both case-change tasks); the gradient-similar word-level pair `reverse_words`/`double_words` sits at 0.24. The residual FM $= 0.13$ nats CE for LoRA-FTN is concentrated on these overlapping pairs; the disjoint pairs retain exactly. This is consistent with Theorem D.1: pairs whose recovered masks are disjoint are retained exactly, and residual forgetting tracks mask overlap empirically; the configurer's job is to make the masks disjoint enough, a job that becomes much easier when the component grid is large enough to give each task its own spatial neighbourhood.

Mask tiling. Figure 4 renders the LoRA-FTN training-time masks across eight random seeds as RGB overlays on the 24×24 component grid: each task is assigned a distinct colour and the per-task binary masks are summed (clipped per-channel). Despite the configurer being cold-started from $m = \mathbf{0}$ at every task, the SmoothKWTA stages (lateral 3×3 kernel, $T = 15$ iterations) consistently group the selected components into compact contiguous blobs. Tasks with disjoint colours are exactly non-interfering by Theorem D.1; the residual co-located colours on gradient-similar task pairs are the source of the residual forgetting reported in Table 3.

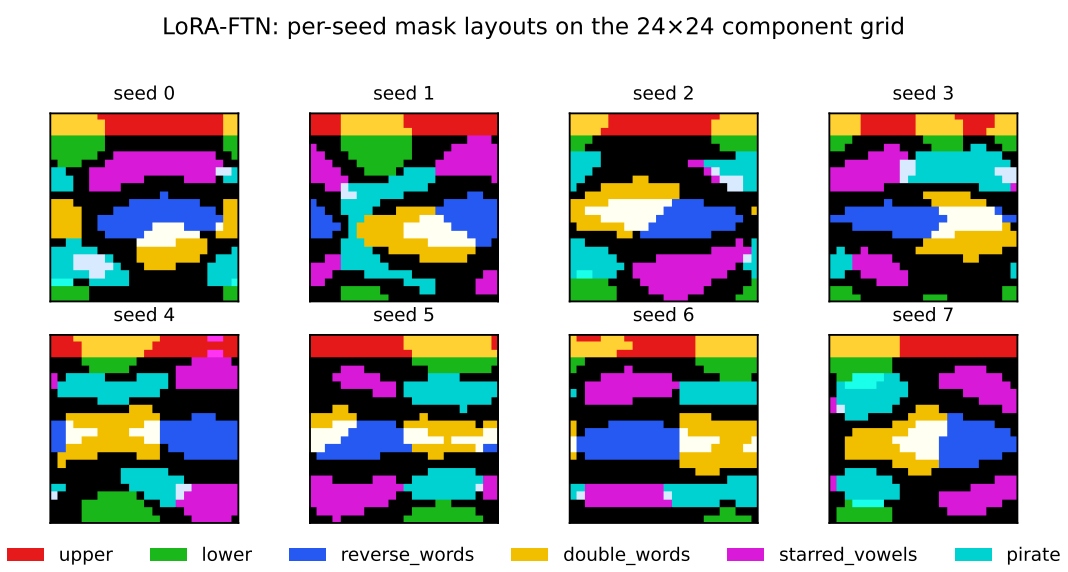


Figure 4. LoRA-FTN mask layouts on the 24×24 component grid for eight random seeds. One colour per task; cells with mixed colour are shared by overlapping masks. The configurer is cold-started from $m = \mathbf{0}$ in every panel, yet repeatedly assigns each task a compact spatial blob, and most task pairs land in disjoint regions across seeds (and are retained exactly).