FUNCTIONAL PROX: DECOUPLING IMPORTANCE ESTIMATION FROM OPTIMIZATION FOR CONTINUAL LEARNING IN SPIKING NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking neural networks (SNNs) offer a promising path toward energy-efficient intelligent computing, yet retaining prior knowledge while continually learning new tasks remains a key challenge for continual adaptation. To mitigate catastrophic forgetting, importance-based regularization methods typically use gradients to estimate parameter importance and consolidate prior knowledge by constraining changes to important parameters. However, the non-differentiability of spike generation makes these estimates depend on the choice of surrogate gradient. Even when network parameters and forward predictions remain unchanged, changing only the backward surrogate can substantially alter hidden-parameter importance rankings or even cause importance values to explode, resulting in misplaced or excessive parameter constraints. To address this issue, we propose **Functional Prox**, a continual learning method that decouples functional importance estimation from surrogate-gradient optimization. Functional Prox estimates importance solely through forward spiking computations, measuring changes in predictive distributions under channel-level interventions to quantify functional contributions. These scores are converted into layer-normalized protection weights and combined with a closed-form, non-expansive proximal update to control constraint magnitudes and parameter displacement. Experiments on CIFAR-100, Tiny-ImageNet, and ImageNet-100 under class-incremental learning (CIL) setting show that Functional Prox significantly outperforms EWC and other representative regularization methods, improving both knowledge retention and new-task learning.

1 INTRODUCTION

Spiking neural networks (SNNs) process information through discrete spikes and their temporal dynamics, offering a promising approach to energy-efficient computation and learning in dynamic environments (Eshraghian et al., 2023; Fang et al., 2023). As SNNs are increasingly considered for sequential learning, acquiring new knowledge while preserving previously learned capabilities becomes a central challenge for long-term adaptation. A range of strategies has been developed to mitigate catastrophic forgetting, including replay, parameter regularization, parameter isolation, and dynamic architectures (Wang et al., 2024). Among these, importance-based regularization estimates the relevance of parameters to previously acquired knowledge and penalizes their subsequent changes, providing a mechanism for knowledge consolidation without replaying past data.

Representative approaches include Elastic Weight Consolidation (EWC), Synaptic Intelligence (SI), and Memory Aware Synapses (MAS). EWC uses a diagonal approximation to the Fisher information matrix to estimate parameter importance for previous tasks (Kirkpatrick et al., 2017). SI accumulates parameter-wise contributions to loss reduction along the optimization trajectory using gradients and parameter updates (Zenke et al., 2017), whereas MAS estimates importance from the gradient sensitivity of the network's output function to parameter perturbations (Aljundi et al., 2018). Despite their different definitions of importance, all three rely on gradients or gradient-dependent learning trajectories. This dependence introduces an additional complication in SNNs. Because the discontinuous spike-generation function does not provide useful derivatives for conventional backpropagation, SNNs are commonly trained by retaining discrete spikes in the forward pass and using continuous surrogate derivatives in the backward pass (Neftci et al., 2019; Eshraghian et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

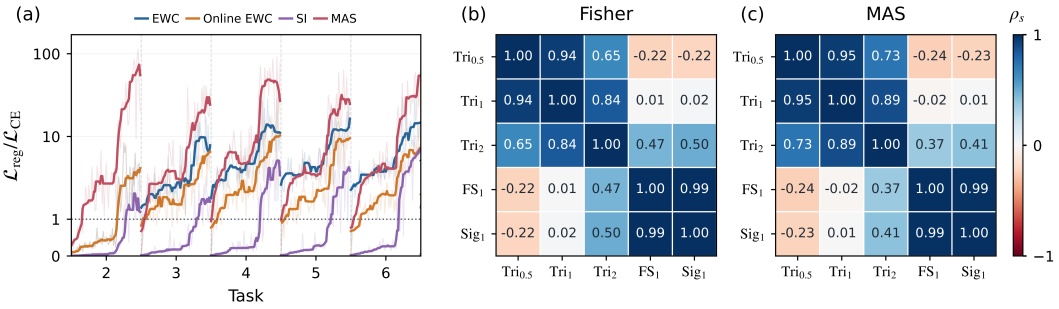


Figure 1: Diagnostics of gradient-based importance in SNNs. (a) Ratio of the regularization loss to current-task loss across tasks. (b,c) Spearman correlations between importance scores obtained at fixed checkpoints with different scoring surrogates for Fisher-type and MAS estimators. Low or negative correlations indicate sensitivity of the protection ranking to the backward rule.

2023). Although surrogate gradients enable effective parameter optimization, their functional form is not uniquely determined by the network's forward computation (Zenke & Vogels, 2021). Different backward surrogates can therefore yield different hidden-layer sensitivities for identical parameters and forward predictions. When these gradients are also used to estimate importance, the resulting protection criterion depends not only on the predictor being preserved but also on the chosen backward surrogate.

Our fixed-checkpoint experiments provide empirical evidence of this dependence, as shown in Figure 1. Holding model parameters, calibration samples, and forward computation fixed, changing only the surrogate used to compute importance gradients produces substantial differences in hidden-layer importance estimates. The resulting rankings can differ markedly and are even negatively correlated in some configurations. Moreover, products of surrogate derivatives across layers and time can strongly amplify importance magnitudes, leading to scale imbalances and potentially numerical instability. These effects extend to subsequent consolidation. Changes in relative importance lead to different protection allocations that cannot be reconciled by adjusting a single global regularization coefficient, whereas inflated scores can impose excessively strong constraints on selected parameters. Such constraints restrict the parameter changes needed to learn subsequent tasks, potentially compromising the balance between knowledge retention and adaptation.

To address these issues, we propose **Functional Prox**, a continual learning method for SNNs that decouples functional importance estimation from surrogate-gradient optimization. At each task boundary, we suppress individual channels and quantify the resulting changes in predictive distributions on current-task calibration samples using the Jensen–Shannon divergence. This forward-only scoring grounds knowledge protection in the measured functional contribution of each channel, independently of the surrogate gradients used for optimization. Functional Prox then converts these scores into sparse channel-level protection through layer-wise selection and normalization, sparsely protecting selected channels and their associated parameters. After the optimizer proposes an update for the current task, the resulting closed-form proximal map shrinks protected parameters toward the reference, with larger weights inducing stronger protection. For fixed reference parameters and nonnegative protection weights, providing a controlled mechanism for parameter consolidation. Our main contributions are summarized as follows:

1. We show that backward-surrogate choice can destabilize SNN importance rankings and magnitudes, motivating surrogate-independent protection.

2. We propose Functional Prox, using forward channel interventions, sparse protection, and proximal consolidation to reduce forgetting without surrogate-dependent importance estimates.

3. We demonstrate consistent improvements over representative importance-based methods on CIFAR-100, Tiny-ImageNet, and ImageNet-100.

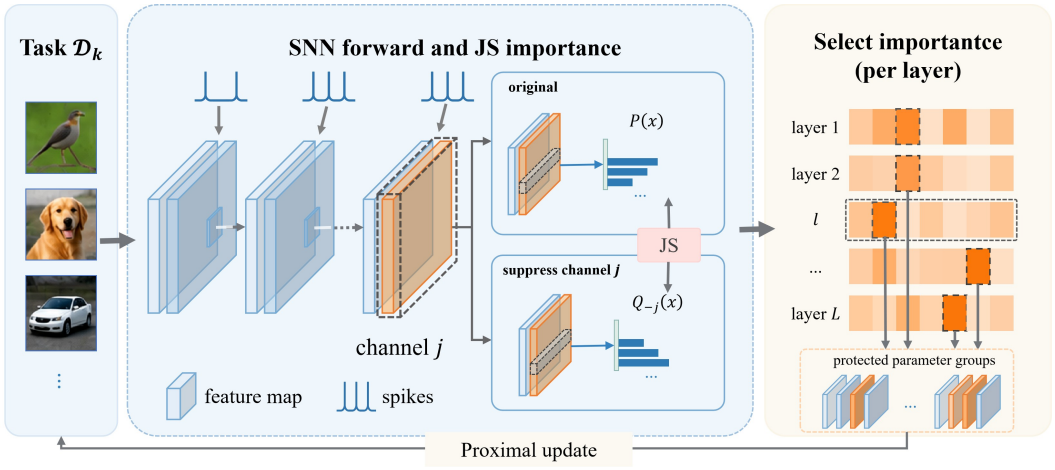


Figure 2: Overview of Functional Prox. Functional importance is estimated by channel interventions and used for sparse protection and proximal consolidation.

2 Related Work

Continual learning. Continual learning addresses forgetting through replay, regularization, and architectural adaptation (Wang et al., 2024). DER combines exemplar replay with output-consistency constraints, but requires additional storage and computation (Buzzega et al., 2020). Parameter isolation within fixed architectures faces capacity limits, whereas dynamic expansion increases model size (Serra et al., 2018; Yoon et al., 2017). Importance-based regularization instead preserves knowledge by selectively constraining parameter updates without an exemplar buffer. EWC, SI, and MAS estimate importance using Fisher information, trajectory-based contributions to loss reduction, and output sensitivity, respectively, with MAS supporting importance updates from unlabeled data (Kirkpatrick et al., 2017; Zenke et al., 2017; Aljundi et al., 2018). Online EWC recursively consolidates historical importance (Schwarz et al., 2018), while AGS-CL combines node importance, and group-sparse penalties to selectively freeze important nodes and reinitialize unimportant ones (Jung et al., 2020). Recent work further improves estimation quality. CSQN captures parameter interactions beyond diagonal curvature through sampled quasi-Newton approximations (Vander Eeckt & Van Hamme, 2025), whereas EWC-DR analyzes output gradients and introduces logit reversal to mitigate gradient vanishing and redundant protection in the identified settings (Liu & Chang, 2026).

Continual learning in spiking neural networks. Continual learning in SNNs explores synaptic plasticity, activity subspaces, and neuronal allocation. Bayesian approaches incorporate weight uncertainty into sequential learning by updating distributions over synaptic weights (Skatchkovsky et al., 2022). HLOP extracts activity subspaces through Hebbian and anti-Hebbian lateral learning, then projects activity traces onto their orthogonal complements to reduce interference with previous tasks (Xiao et al., 2024). HLML-SNN combines sample-level Hebbian adaptation with task-level meta-learning to support rapid adaptation and cross-task consolidation (Xu et al., 2026). For architectural adaptation, SCA-SNN uses task similarity to regulate neuron reuse and expansion, while ALADE-SNN combines dynamic expansion with adaptive logit alignment to address output biases across old and new tasks (Han et al., 2025; Ni et al., 2025). RTK-WTA couples temporal activity traces with probabilistic top-k competition, introducing controlled randomness while retaining temporal selectivity to reduce representational overlap and improve robustness to perturbations (Shen et al., 2026).

3 Methods

This section introduces Functional Prox (as shown in Figure 2) and its key components for continual learning in SNNs.

3.1 PRELIMINARIES

Class-incremental learning. Consider a sequence of K tasks $\{\mathcal{D}_k\}_{k=1}^{K}$ with disjoint class sets $\mathcal{C}_k$. During task k, the learner has access only to current-task training data, while evaluation requires prediction over all classes observed so far without task identity. Using a single reference point, quadratic importance regularization takes the form

$$\mathcal{J}_k(\theta) = \mathcal{L}_k(\theta) + \lambda \sum_i \Omega_{k-1,i} \left(\theta_i - \theta^*_{k-1,i}\right)^2, \tag{1}$$

where $\mathcal{L}_k$ is the current-task loss, θ^*_{k-1} is the reference saved at the previous task boundary, $\Omega_{k-1,i}$ denotes the retained importance of parameter i, and λ controls the overall consolidation strength. Representative importance-based methods differ mainly in how $\Omega_{k-1,i}$ is estimated. EWC uses a Fisher-type sensitivity score (Kirkpatrick et al., 2017), SI accumulates gradient–parameter-update contributions along the optimization trajectory (Zenke et al., 2017), and MAS measures the sensitivity of the network output to parameter perturbations (Aljundi et al., 2018). Although their formulations differ, these estimators all involve backward derivatives or gradient-dependent trajectories.

Spiking neurons and surrogate gradients. We consider an SNN with L spiking layers and T simulation steps. The neuronal dynamics are represented abstractly as

$$u^l_t = \mathcal{F}(u^l_{t-1}, s^{l-1}_t; \theta), \qquad s^l_t = H(u^l_t - \vartheta), \tag{2}$$

where u^l_t denotes the membrane state, s^l_t is the binary spike output, ϑ is the firing threshold, and $\mathcal{F}$ represents the membrane and reset dynamics, such as those of a leaky integrate-and-fire (LIF) neuron (Gerstner et al., 1945). Since the threshold function $H(a) = \mathbb{I}[a \geq 0]$ is non-differentiable at the threshold and has zero derivative elsewhere, surrogate-gradient training replaces its backward derivative with

$$\widetilde{\frac{\partial s^l_t}{\partial u^l_t}} = \psi_\sigma(u^l_t - \vartheta), \tag{3}$$

where ψ_σ denotes the chosen surrogate derivative. Changing σ leaves the forward spikes and predictions unchanged for fixed parameters and inputs, but changes the backward differentiation rule.

To make this dependence explicit, let

$$\widetilde{J}_\sigma(x) = \frac{\widetilde{\partial}_\sigma z_\theta(x)}{\partial \theta} \tag{4}$$

denote the surrogate Jacobian of the logits. For cross-entropy loss, the parameter gradient can be written as

$$\widetilde{\nabla}_\sigma \mathcal{L} = \widetilde{J}_\sigma(x)^\top \big(p_\theta(x) - y\big), \tag{5}$$

where $p_\theta(x) = \mathrm{softmax}(z_\theta(x))$. At a fixed checkpoint, changing only the scoring surrogate leaves the prediction $p_\theta(x)$ and the output residual $p_\theta(x) - y$ unchanged, while $\widetilde{J}_\sigma(x)$ can vary. Gradient-based importance estimates therefore inherit this dependence. For example, a Fisher-type score can be written as

$$\widehat{F}^{(\sigma)}_{k,i} = \frac{1}{N_k} \sum_{n=1}^{N_k} \left[\widetilde{g}_{\sigma,i}(x_n, y_n)\right]^2, \tag{6}$$

where $\widetilde{g}_\sigma$ is the surrogate gradient. MAS similarly depends on output gradients, while SI accumulates gradient-dependent quantities throughout training.

Propagation of surrogate dependence. The effect of the surrogate becomes more pronounced for hidden parameters because backward signals contain products of surrogate derivatives across layers and time. For a feedforward stack, define

$$D^l_\sigma = \mathrm{diag}\left(\psi_\sigma(u^l - \vartheta)\right). \tag{7}$$

The hidden error signal satisfies

$$\delta^l_\sigma = D^l_\sigma (W^{l+1})^\top \delta^{l+1}_\sigma, \tag{8}$$

and can therefore be expanded as

$$\delta^l_\sigma = D^l_\sigma (W^{l+1})^\top D^{l+1}_\sigma \cdots (W^L)^\top D^L_\sigma (W^{\mathrm{out}})^\top r, \tag{9}$$

where r denotes the output residual. Thus, even when the forward computation is fixed, changing the surrogate modifies multiple derivative factors along the backward path. For multi-step SNNs, temporal state transitions introduce additional surrogate-dependent Jacobian products, further affecting the gradients propagated through time.

A simple scaling example illustrates the resulting magnitude sensitivity. If the surrogate derivative is rescaled as $\psi_{\sigma'} = c\psi_\sigma$ with $c > 0$, then for a single-step L-layer chain,

$$\widetilde{\nabla}_{W^l,\sigma'}\mathcal{L} = c^{L-l+1}\widetilde{\nabla}_{W^l,\sigma}\mathcal{L}. \tag{10}$$

Accordingly, Fisher-type importance scales with the square of this factor. More generally, changes in surrogate shape or width can affect different units non-uniformly, altering not only importance magnitudes but also parameter rankings. This motivates estimating functional importance without relying on backward surrogate gradients.

3.2 Functional Prox

Functional importance from channel interventions. At the end of task k, we fix the learned parameters θ_k^* and sample a calibration set $\mathcal{B}_k = \{x_n\}_{n=1}^{N_k}$ from the current task. For each candidate channel j, we suppress its transmitted spike outputs across all spatial locations and simulation steps while keeping the model parameters and neuronal dynamics unchanged. Let s_t^l denote the corresponding spike tensor. The intervened output is defined as

$$s_{t,c,h,w}^{l,(-j)} = \begin{cases} 0, & c = j, \\ s_{t,c,h,w}^{l}, & c \neq j, \end{cases} \qquad t = 1, \ldots, T. \tag{11}$$

We then compare the predictive distributions of the intact and intervened networks,

$$\begin{aligned} P_k(x) &= \mathrm{softmax}(z_{\theta_k^*}(x)), \\ Q_{k,-j}(x) &= \mathrm{softmax}(z_{\theta_k^*}^{(-j)}(x)). \end{aligned} \tag{12}$$

The functional importance of channel j is measured by the average Jensen–Shannon (JS) divergence between the two distributions,

$$D_{k,j} = \frac{1}{N_k}\sum_{n=1}^{N_k} \mathrm{JS}\big(P_k(x_n), Q_{k,-j}(x_n)\big), \tag{13}$$

where

$$\mathrm{JS}(P,Q) = \frac{1}{2}\,\mathrm{KL}(P\|M) + \frac{1}{2}\,\mathrm{KL}(Q\|M), \qquad M = \frac{1}{2}(P+Q). \tag{14}$$

A larger $D_{k,j}$ indicates that suppressing channel j induces a larger change in the model prediction and therefore that the channel makes a greater functional contribution to the current predictor. Because this score is computed entirely from forward predictions, it is independent of the backward surrogate under fixed parameters and evaluation conditions.

Layer-normalized sparse protection. We convert the functional score into an allocation score

$$a_{k,j} = \left(\frac{D_{k,j}}{\max(r_{k,j}, r_{\min})^\alpha}\right)^{p/2}, \tag{15}$$

where $r_{k,j}$ is the mean firing rate of channel j, $r_{\min} > 0$ prevents numerical instability for rarely active channels, $\alpha \geq 0$ controls firing-rate adjustment, and $p > 0$ controls the concentration of the resulting importance distribution.

For each layer, we retain the top-ρ fraction of channels according to $a_{k,j}$ and denote the selected set by S_k^l. The corresponding channel weights are normalized within the selected set:

$$w_{k,j} = \begin{cases} \dfrac{a_{k,j}}{\sum_{q\in S_k^l} a_{k,q}}, & j \in S_k^l, \\ 0, & j \notin S_k^l. \end{cases} \tag{16}$$

Thus, the selected channels in each layer share a unit protection budget, while ρ controls the sparsity of the allocation. Each selected channel is associated with a parameter group containing the parameters responsible for producing that channel. The channel weight $w_{k,j}$ is assigned to all parameters, yielding the task-specific importance coefficients $\widehat{\Omega}_{k,i}$. Parameters associated with unselected channels receive zero weight.

To retain information from previous tasks without storing task-specific importance statistics, we update the historical coefficients online:

$$\Omega_{k,i} = \gamma \Omega_{k-1,i} + \widehat{\Omega}_{k,i}, \tag{17}$$

where $\gamma \in [0, 1)$ controls the decay of past importance. The resulting Ω_k and parameter reference θ^*_k are fixed while learning task $k + 1$. This accumulation preserves protection for parameters that were important to earlier tasks while gradually reducing the influence of older estimates.

Proximal consolidation. During task k, we keep θ^*_{k-1} and Ω_{k-1} fixed. At optimization step m, the current-task optimizer first produces a proposal $\widetilde{\theta}^{(m+1)}$. We then apply a quadratic proximal update

$$\theta^{(m+1)} = \operatorname*{argmin}_{\theta} \left\{ \frac{\|\theta - \widetilde{\theta}^{(m+1)}\|_2^2}{2\eta_m} + \lambda \sum_i \Omega_{k-1,i} \left(\theta_i - \theta^*_{k-1,i}\right)^2 \right\}, \tag{18}$$

where $\eta_m > 0$ is the proximal step size. Because the objective is separable across parameters, the solution is available in closed form:

$$\theta_i^{(m+1)} = \theta^*_{k-1,i} + \frac{\widetilde{\theta}_i^{(m+1)} - \theta^*_{k-1,i}}{1 + 2\eta_m \lambda \Omega_{k-1,i}}. \tag{19}$$

For $\Omega_{k-1,i} = 0$, the optimizer proposal is unchanged. For protected parameters, the update contracts the proposal toward the previous-task reference with factor

$$c_{m,i} = \frac{1}{1 + 2\eta_m \lambda \Omega_{k-1,i}} \in (0, 1]. \tag{20}$$

Thus, larger importance weights induce stronger consolidation while the proximal map remains non-expansive for fixed nonnegative coefficients. By contrast, an explicit gradient step on the same quadratic penalty gives

$$c_{m,i}^{\text{explicit}} = 1 - 2\eta_m \lambda \Omega_{k-1,i}, \tag{21}$$

whose magnitude can exceed one when the penalty becomes large. The proximal formulation therefore provides a stable way to enforce strong, importance-weighted constraints without directly coupling their numerical scale to the stability of an explicit regularization step (Jung et al., 2020).

4 Experiments

In this section, we evaluate Functional Prox on class-incremental learning and analyze the contribution of its key components through ablation studies.

4.1 Experimental Setup

We evaluate replay-free class-incremental learning on CIFAR-100 (Krizhevsky et al., 2009), Tiny-ImageNet (Le et al., 2015), and ImageNet-100 (Russakovsky et al., 2015). CIFAR-100 contains 100 classes of images, with 500 training and 100 test samples per class. Tiny-ImageNet contains 200 classes with 500 training and 50 evaluation samples per class. ImageNet-100 consists of 100 classes selected from ImageNet, with 1,300 training and 50 validation images per class. Following Liu & Chang (2026), we consider both big-start and equally-split class-incremental protocols. For big-start evaluation, CIFAR-100 and ImageNet-100 begin with a 50-class base task, followed by either $M = 5$ increments of 10 classes or $M = 10$ increments of 5 classes. Tiny-ImageNet begins with 100 classes, followed by either five 20-class increments or ten 10-class increments. For equally-split evaluation, all classes are uniformly divided into $K \in \{5, 10, 20\}$ tasks. This results in 20, 10, or 5 classes per task for CIFAR-100 and ImageNet-100, and 40, 20, or 10 classes per task for Tiny-ImageNet.

Table 1: CIL results on CIFAR-100 with SNN implementations of all methods.

	Big-start				Equally-split					
	$M = 5$		$M = 10$		$K = 5$		$K = 10$		$K = 20$	
Method	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}
EWC	14.66	30.04	11.25	26.90	15.96	29.11	8.64	18.49	5.73	16.28
Online EWC	26.37	43.24	25.30	38.44	22.18	39.72	16.33	30.17	10.06	23.74
SI	17.56	36.82	11.89	26.55	14.35	30.59	9.73	22.68	7.58	17.03
MAS	32.24	47.86	28.72	41.55	25.02	35.16	20.73	37.19	13.97	28.51
EWC-DR	49.29	60.58	43.06	59.91	41.43	61.28	26.02	41.90	17.91	33.24
Functional Prox	**52.71**	**62.94**	**46.88**	**60.51**	**43.82**	**62.96**	**29.37**	**44.41**	**21.40**	**37.72**

Our implementation is based on PyCIL (Zhou et al., 2021). We use a spiking ResNet-18 backbone (He et al., 2016) with leaky integrate-and-fire (LIF) neurons (Gerstner et al., 1945), $T = 4$ simulation steps, and the ZIF surrogate function. All methods are trained from scratch using SGD with the same optimization schedule. The base task is trained for 200 epochs, while subsequent tasks are trained for 180 epochs, both with an initial learning rate of 0.1. For Functional Prox, we set the protection fraction to $\rho = 0.1$, use $N_k = 32$ calibration samples at each task boundary, set the consolidation strength to $\lambda = 0.6$, and use historical decay $\gamma = 0.5$.

After each task k, we evaluate the model on all classes observed so far and denote the resulting accuracy by A_k. We report the final accuracy $A_{\text{last}} = A_K$ and the average incremental accuracy

$$A_{\text{avg}} = \frac{1}{K} \sum_{k=1}^{K} A_k. \tag{22}$$

The former reflects performance at the end of the task sequence, while the latter summarizes performance throughout continual learning. All results are reported in percentages, with differences expressed in percentage points (pp).

4.2 Class-Incremental Learning Results

We compare Functional Prox with EWC (Kirkpatrick et al., 2017), Online EWC (Schwarz et al., 2018), SI (Zenke et al., 2017), MAS (Aljundi et al., 2018), and EWC-DR (Liu & Chang, 2026). All methods use the same SNN backbone and task-training schedule.

On CIFAR-100, Functional Prox outperforms all baselines on both metrics in every configuration. Compared with EWC-DR, the gains in A_{last} are 3.42 and 3.82 percentage points (pp) under the two big-start settings, and 2.39, 3.35, and 3.49 pp under the 5-, 10-, and 20-task equally-split settings, respectively. For A_{avg}, the corresponding gains are 2.36 and 0.60 pp under big-start, and 1.68, 2.51, and 4.48 pp under equally-split evaluation. The improvement remains clear as the task sequence becomes longer, particularly for average incremental accuracy.

On ImageNet-100, Functional Prox achieves the best performance in four of the five configurations. Under big-start with $M = 5$, it improves over EWC-DR by 0.80 pp in A_{last} and 1.28 pp in A_{avg}. Under equally-split evaluation, the gains over EWC-DR are 3.41/2.98 pp, 1.19/2.99 pp, and 0.29/3.43 pp for $K = 5$, 10, and 20, respectively. The only exception is big-start with $M = 10$, where EWC-DR is higher by 0.56 pp in final accuracy and 1.32 pp in average accuracy.

On Tiny-ImageNet, Functional Prox achieves the highest final and average accuracy in all five configurations. Compared with EWC-DR, the gains in A_{last} range from 0.23 to 3.90 pp, while the gains in A_{avg} range from 0.46 to 2.54 pp. The largest gain in final accuracy appears under the 20-task equally-split setting, where Functional Prox reaches 15.62%, compared with 11.72% for EWC-DR.

Across the three benchmarks, Functional Prox achieves strong performance under both big-start and equally-split protocols. The gains are generally maintained as the number of tasks increases, showing that the method remains effective across different class-incremental settings.

Table 2: CIL results on ImageNet-100 with SNN implementations of all methods.

	Big-start				Equally-split					
	$M = 5$		$M = 10$		$K = 5$		$K = 10$		$K = 20$	
Method	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}
EWC	9.10	17.53	7.38	15.88	19.67	31.49	10.56	21.45	4.94	14.67
Online EWC	21.78	37.52	19.91	33.59	23.64	34.91	15.71	26.78	10.14	18.52
SI	7.13	16.68	4.36	10.25	12.10	25.97	7.81	18.55	4.02	12.13
MAS	22.72	36.09	14.43	26.85	19.63	31.79	10.95	24.54	8.74	16.40
EWC-DR	64.14	73.28	**57.43**	**67.91**	51.19	66.27	33.94	51.77	14.89	30.16
Functional Prox	**64.94**	**74.56**	56.87	66.59	**54.60**	**69.25**	**35.13**	**54.76**	**15.18**	**33.59**

Table 3: CIL results on Tiny-ImageNet with SNN implementations of all methods.

	Big-start				Equally-split					
	$M = 5$		$M = 10$		$K = 5$		$K = 10$		$K = 20$	
Method	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}	A_{last}	A_{avg}
EWC	8.40	17.62	5.72	14.21	17.96	28.63	10.53	22.96	5.74	15.13
Online EWC	25.32	37.16	17.95	30.27	22.93	34.09	16.75	26.42	10.91	18.36
SI	7.84	16.23	4.95	14.81	12.34	26.96	7.02	18.55	4.88	15.03
MAS	24.19	38.04	14.26	26.43	18.53	30.46	10.51	22.79	7.53	16.23
EWC-DR	34.99	44.27	29.53	38.54	27.14	35.94	19.68	30.75	11.72	26.43
Functional Prox	**36.93**	**46.81**	**29.76**	**39.00**	**27.80**	**36.62**	**21.47**	**33.12**	**15.62**	**27.46**

4.3 Ablation Studies

We conduct ablation studies on CIFAR-100 under the equally-split $K = 20$ setting. Unless otherwise specified, all variants use the same backbone, training schedule, calibration size $N_k = 32$, protection fraction $\rho = 0.1$, consolidation strength $\lambda = 0.6$, and historical decay $\gamma = 0.5$. The full Functional Prox model achieves 21.40% A_{last} and 37.72% A_{avg}.

Intervention granularity. We first examine the granularity used for functional importance estimation. Channel–time intervention suppresses one channel at a single simulation step, whereas channel intervention suppresses the same channel throughout the full simulation. As shown in Table 4, channel intervention improves A_{last} from 19.90% to 21.40% and A_{avg} from 35.19% to 37.72%, corresponding to gains of 1.50 and 2.53 pp, respectively.

Channel-level intervention is also substantially more efficient. With $T = 4$, the candidate space is reduced from 17.6k channel–time candidates to 4.4k channel candidates, while the runtime overhead decreases from 17.3% to 4.1%. Finer weight- and neuron-level interventions involve more than one million candidates, making exhaustive evaluation computationally prohibitive. These results support channel-level intervention as a practical granularity for functional importance estimation.

Protection allocation. We next examine how functional importance is converted into protection weights. As shown in Table 5, assigning uniform weights to the selected channels reduces A_{last} by 12.84 pp and A_{avg} by 10.28 pp, showing that the relative importance among selected channels is important for determining protection strength.

Dense protection ($\rho = 1$) decreases final and average accuracy by 2.93 and 4.26 pp, respectively. This shows the benefit of concentrating the protection budget on a sparse subset of functionally important channels rather than distributing it across all channels.

Replacing layer-wise normalization with global normalization causes the largest degradation among the allocation variants, reducing A_{last} by 13.80 pp and A_{avg} by 14.10 pp. This indicates that balancing protection across layers is important when functional scores differ substantially in scale.

Table 4: Intervention-granularity comparison on CIFAR-100 with $K = 20$. "–" denotes computationally prohibitive settings.

Intervention	Candidates	Overhead	A_{last}	A_{avg}
Weight	12,548k	–	–	–
Neuron–time	5,505.0k	–	–	–
Neuron	1,376.2k	–	–	–
Channel–time	17.6k	17.3%	19.90	35.19
Channel	4.4k	4.1%	**21.40**	**37.72**

Table 5: Component and scoring–update ablations on CIFAR-100 with $K = 20$.

Variant	Consolidation	A_{last}	A_{avg}
Functional Prox	Proximal	**21.40**	**37.72**
Uniform weights on selected channels	Proximal	8.56	27.44
Dense protection ($\rho = 1$)	Proximal	18.47	33.46
Global normalization	Proximal	7.60	23.62
No firing-rate adjustment ($\alpha = 0$)	Proximal	21.37	37.45
Functional scores	Explicit	14.14	29.42
Fisher-type scores	Proximal	12.71	25.94
Fisher-type scores	Explicit	4.94	13.67
No consolidation ($\lambda = 0$)	None	4.24	12.19

In contrast, removing firing-rate adjustment has only a minor effect. Setting $\alpha = 0$ results in 21.37% A_{last} and 37.45% A_{avg}, close to the full method.

Importance scoring and consolidation. We further separate the effects of importance estimation and consolidation. With functional scores fixed, replacing explicit regularization with proximal consolidation improves A_{last} from 14.14% to 21.40% and A_{avg} from 29.42% to 37.72%, corresponding to gains of 7.26 and 8.30 pp. With proximal consolidation fixed, replacing Fisher-type scores with functional scores improves final and average accuracy by 8.69 and 11.78 pp, respectively. This shows that both functional importance estimation and proximal consolidation contribute substantially to the overall performance. Finally, removing consolidation entirely ($\lambda = 0$) reduces A_{last} to 4.24% and A_{avg} to 12.19%. Together, these ablations show that functional scoring, sparse layer-wise protection, and proximal consolidation each play an important role in Functional Prox.

5 Conclusion

This work investigates the effect of surrogate gradients on parameter importance estimation in continual learning for SNNs and proposes Functional Prox to decouple importance estimation from surrogate-gradient optimization. Experiments on CIFAR-100, Tiny-ImageNet, and ImageNet-100 show that Functional Prox performs well across different CIL task partitions and sequence lengths. The ablation studies further verify the contributions of functional importance scoring, sparse protection allocation, and proximal consolidation to the overall performance. Functional Prox also has several limitations. Its importance estimation depends on the calibration samples available at task boundaries and on the predefined intervention location, while single-channel interventions cannot fully capture interactions and redundancy among channels. In addition, the number of channel-level interventions grows approximately linearly with the number of candidate channels, which may introduce additional computational overhead for larger networks. Future work can further improve the method in several directions. First, more efficient joint or structured intervention strategies could be explored to model inter-channel dependencies while reducing evaluation cost. Second, adaptive calibration-sample selection and historical importance update mechanisms could be investigated to improve stability and scalability over longer task sequences and larger models.

REFERENCES

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *European conference on computer vision*, pp. 144–161. Springer, 2018.

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.

Jason K Eshraghian, Max Ward, Emre O Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9):1016–1054, 2023.

Wei Fang, Yanqi Chen, Jianhao Ding, Zhaofei Yu, Timothée Masquelier, Ding Chen, Liwei Huang, Huihui Zhou, Guoqi Li, and Yonghong Tian. Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances*, 9(40):eadi1480, 2023.

Wulfram Gerstner, Werner M Kistler, Richard Naud, and Liam Paninski. Neuronal dynamics. *American history*, 1861(1900), 1945.

Bing Han, Feifei Zhao, Yang Li, Qingqun Kong, Xianqi Li, and Yi Zeng. Similarity-based context aware continual learning for spiking neural networks. *Neural Networks*, 184:107037, 2025.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Sangwon Jung, Hongjoon Ahn, Sungmin Cha, and Taesup Moon. Continual learning with node-importance based adaptive group sparse regularization. *Advances in neural information processing systems*, 33:3647–3658, 2020.

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*, 114 (13):3521–3526, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann Le, Xuan Yang, et al. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Xuan Liu and Xiaobin Chang. Elastic weight consolidation done right for continual learning. *arXiv preprint arXiv:2603.18596*, 2026.

Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

Wenyao Ni, Jiangrong Shen, Qi Xu, and Huajin Tang. Alade-snn: Adaptive logit alignment in dynamically expandable spiking neural networks for class incremental learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 19712–19720, 2025.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, 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*, pp. 4528–4537. PMLR, 2018.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International conference on machine learning*, pp. 4548–4557. PMLR, 2018.

Jiangrong Shen, Liang Zhao, Qi Xu, Yuqi Yang, Liangjun Chen, Gang Pan, and Badong Chen. Robust selective activation with randomized temporal k-winner-take-all in spiking neural networks for continual learning. In *International Conference on Learning Representations*, volume 2026, pp. 131779–131797, 2026.

Nicolas Skatchkovsky, Hyeryung Jang, and Osvaldo Simeone. Bayesian continual learning via spiking neural networks. *Frontiers in Computational Neuroscience*, 16:1037976, 2022.

Steven Vander Eeckt and Hugo Van Hamme. Continual learning with quasi-newton methods. *Ieee Access*, 13:47485–47499, 2025.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5362–5383, 2024.

Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Di He, and Zhouchen Lin. Hebbian learning based orthogonal projection for continual learning of spiking neural networks. In *International Conference on Learning Representations*, volume 2024, pp. 16271–16293, 2024.

Jiangshuai Xu, Peiyun Xue, Jiacheng Song, Xuhui Huang, and Qingshan Hou. Hlml-snn: Fast continual learning in spiking neural networks achieved via hebbian learning-driven meta-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 27260–27268, 2026.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

Friedemann Zenke and Tim P Vogels. The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *Neural computation*, 33(4):899–925, 2021.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pp. 3987–3995. Pmlr, 2017.

Da-Wei Zhou, Fu-Yun Wang, Han-Jia Ye, and De-Chuan Zhan. Pycil: a python toolbox for class-incremental learning. *arXiv preprint arXiv:2112.12533*, 2, 2021.

GENERATIVE AI USAGE DISCLOSURE

Generative AI tools, including large language models (LLMs), were used in this work primarily to assist with manuscript preparation and experimental code development.

Manuscript preparation. LLMs were used to assist with drafting, rewriting, polishing, and improving the clarity and readability of parts of the manuscript. All AI-assisted text was reviewed and revised by the authors, who determined and verified the scientific claims, interpretations, and conclusions.

Experimental code development. LLMs were used to support the development of selected parts of the experimental code, including suggesting implementations, assisting with debugging, and improving existing scripts for data processing, model inference, and evaluation. All code used in the experiments was reviewed, tested, and verified by the authors before producing the reported results.

Responsibility. The authors reviewed all AI-assisted content and take full responsibility for the final manuscript, implementation, experiments, results, and conclusions of this work.