NPCL: Neural Processes for Uncertainty-Aware Continual Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Continual learning (CL) aims to train deep neural networks (DNNs) efficiently on streaming data while limiting the forgetting caused by new tasks. However, learning transferable knowledge with less interference between tasks is difficult, and real-world deployment of CL models is limited by their inability to measure predictive uncertainties. To address these issues, we propose handling CL tasks with neural processes (NPs), a class of meta-learners that encode different tasks into probabilistic distributions over functions all while providing reliable uncertainty estimates. Specifically, we propose an NP-based CL approach (NPCL) with task-specific modules arranged in a hierarchical latent variable model. We tailor regularizers on the learned latent distributions to alleviate forgetting. We then use uncertainty estimation capabilities of NPCL to handle the fundamental CL challenge of task head inference. Our experiments show that NPCL outperforms previous CL approaches. We validate the effectiveness of uncertainty estimation in NPCL for identifying novel data and evaluating instance-level model confidence.

1 Introduction

Continual learning (CL) aims to help deep neural networks (DNNs) learn from a stream of non-stationary tasks by retaining the previously acquired knowledge [48, 32]. To achieve this, CL agents target alleviating the *catastrophic forgetting* issue with restricted computational and memory costs [37]. This requires balancing the plasticity for new knowledge with the stability for old [33].

To avoid forgetting in CL, experience replay (ER) methods [27, 6] are one effective way to train DNNs on a memory buffer with a subset of the past tasks' experiences. Other than the ER methods, many regularization-based approaches have also been proposed to penalize the forgetting on the DNNs' parametric [27] or representation spaces [5, 4]. However, these may still suffer from interference due to the regularization on the entire parameter space [7]. To address this, parameter isolation methods [47, 30] define task-specific training components but are usually confined to task incremental CL setups [42]. It is thus challenging for CL agents to maintain transferable and shareable knowledge. Furthermore, a hurdle to the real-world deployment of CL agents is their inability to measure predictive uncertainties. This, like other autonomous learning agents, keeps them from safety critical applications [28].

To tackle the above issues, we propose to explore CL models using neural processes (NPs) [12, 13], a class of meta-learners that model tasks as data generating functions from a stochastic process. NPs learn a prior over functions by marginalizing over a set of data points, or *context*, thus enabling rapid adaptation to new observations through inference on functions. Additionally, their probabilistic nature endows them with reliable uncertainty quantification capabilities [13, 24]. Our motivations to explore NPs for CL are thus two-fold. First, NPs exploit Bayes' theorem which naturally enables CL through sequential posterior construction. Namely, NPs perform inference over the function

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

space by learning context-based priors which are updated to posteriors upon observing (additional) *targets*. Second, NPs meta-learn input correlations through a latent variable. For CL, this could be the key to meta-learn knowledge transfer across correlated tasks. However, NPs cannot directly handle CL tasks given: (a) the reliance on a single global latent leads to suboptimal modeling of complex CL signals where multiple correlated tasks could occur simultaneously, (b) NPs still need to handle the forgetting of past task correlations arising from the non-static data stream.

To address the above desiderata, we propose Neural Processes for Continual Learning (NPCL), a hierarchical latent variable model with a global latent to capture inter-task correlation and task-specific latents for finer knowledge. Fig. 1 shows NPCL exploiting functional correlation among current and past task training samples of ER. The drift of global and past task-specific distributions away from their original forms serve as the major aspect of forgetting in NPCL. We thus propose to regularize these towards their old forms and show the merits of it over typical parameter-based regularization. We then leverage the uncertainty encoded by NPCL for the aforesaid CL challenge of task head inference. To this end, we propose using entropy as an uncertainty quantification metric (UQM). NPCL outperforms previous probabilistic CL models and delivers better or comparable results than state-of-the-art deterministic CL methods, which usually have an edge over their probabilistic counterparts in terms of accuracy. To study the further usages of NPCL's uncertainty estimation, we show its out-of-the-box readiness for novel data detection and instance-level confidence evaluation [16]. Lastly, we list the key limitations of NPCL as an attempt to lay further solid directions for uncertainty-aware CL.

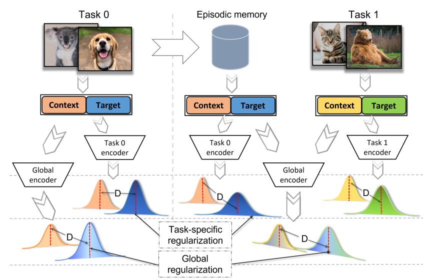


Figure 1: **Neural Processes for Continual Learning**: each training step involves minimizing the distance D between the context-based prior and the target-based posterior, alongside regularizing the task-specific and global distributions towards their old forms.

2 Related Work

Continual Learning (CL). CL methods address catastrophic forgetting through three major approaches: (a) *Regularization-based* methods penalize changes in a model's important weights for previous tasks; e.g., Elastic Weight Consolidation (EWC) [27], Synaptic Intelligence (SI) [48], etc. (b) *Parameter Isolation-based* methods partition the network's parameters to specialize on individual tasks; e.g., Yan et al. [46] using variational Bayesian sparsity priors to reserve model capacity for future tasks, Douillard et al. [8] learning task-specific tokens for Transformers, etc. (c) *Replay-based* methods use an episodic memory to preserve a fraction of the past tasks' experience, and use these to prevent forgetting while learning on new tasks; e.g., experience replay (ER) [6] storing past inputs, and dark experience replay (DER) [4] storing past logits. Our method uses (a) via regularization of distributions, (b) via task-specific latent heads, and (c) via replay of past task inputs and distributions.

Neural Processes (NPs). NPs were introduced to meta-learn a family of data-generating functions through their deterministic [12] and / or latent summaries [13]. Attentive NPs (ANPs) [24] replaced the averaging operation in NPs with a dot-product attention [43] to enhance their expressivity. (A)NPs rely on a global latent that limits their ability to model observations from multiple functions. Recent works address this through local latents that model fine-grained correlation among a subset of the observations [45]. In particular, our work is inspired by multi-task processes (MTPs) [23] that model multiple tasks owing to the hierarchy of task-specific latents conditioned on a global latent. However, MTPs have more relaxed constraints than CL because: (a) MTPs are not trained sequentially on tasks and are thus free of forgetting; (b) at test time, MTPs assume each input to be mapped to all tasks and thus bypass the issue of task head inference pervading class-incremental learning.

Besides, the added complexity of variational inference has limited NP applications to mostly toy regression tasks [22]. The potential of NPs for large-scale classification tasks thus remains largely unexplored, if not untouched. Wang et al. [44], for instance, leverage the predictive uncertainties of NPs to decide on pseudolabels for unlabeled data in semi-supervised classification. We, therefore, use NPs for CL because of a number of their intrinsic properties including principled Bayesian learning, uncertainty estimation, and easy integration with preexisting CL methods like ER.

3 Preliminaries: Neural Processes

NPs [12, 13] meta-learn a task t as the mapping $F_*^t : X^t \to Y^t$. F_*^t generates data $\{(x^t, y^t)\} = (X^t, Y^t) \sim \mathcal{D}^t$ constituting of: (a) a context set $|\mathcal{C}^t| = m$ that offers a prior, and (b) a target set $|\mathcal{T}^t| = m + n$ that contains additional samples to compute the posterior over the full observations. NPs learn the Gaussian priors and posteriors using a neural network $F^t_{[\phi;\theta]} \approx F^t_*$, where ϕ and θ parameterize an encoder q and a decoder p, respectively. This involves deriving a global variable z^G to estimate the prior $p(z^G|\mathcal{C}^t; \phi)$, and then maximizing the marginal likelihood $p(Y^t_{\mathcal{T}}|\mathcal{C}^t, X^t_{\mathcal{T}}; \theta)$:

$$p(Y^t_{\mathcal{T}}|X^t_{\mathcal{T}}, \mathcal{C}^t) = \int p(Y^t_{\mathcal{T}}|X^t_{\mathcal{T}}, z^G)p(z^G|\mathcal{C}^t)dz^G, \tag{1}$$

where $p(Y^t_{\mathcal{T}}|X^t_{\mathcal{T}}, z^G) = \prod_{i=1}^{m+n} p(y^t_i|x^t_i, z^G)$ is the generative likelihood. In a CL setup, memorizing the task prior $p(z^G|\mathcal{C}^t; \phi)$ can help NPs avoid forgetting the *t-th* task. Our aim behind enabling NP for CL is to seek a trade-off to preserve such task priors while sharing the parameters among tasks.

4 Continual Learning with Neural Processes

A CL setup considers $\mathcal{D}^t$ from $0 \le t \le T - 1$ sequentially arriving tasks. Using cross-entropy (CE) as the classification loss l, the CL objective for the task t involves minimizing:

$$\mathcal{L}^t_{CE} = \mathbb{E}_{(x,y)\sim\mathcal{D}^t}\ l(F_{[\phi;\theta]}(x), y) \tag{2}$$

on all $[0, t]$ seen tasks. Achieving Eq. (2) is challenging in real-world scenarios where the previous datasets can be unavailable due to constraints on privacy, storage, etc. To bypass this, several CL approaches employ *experience replay* (ER) where a small episodic memory $\mathcal{M}$ is updated periodically to store and revisit minibatches of past experiences $(x^t, y^t) \sim \mathcal{M}$ for a task t [6, 32]. In this work, we rely on reservoir sampling [6] for a task boundary-agnostic updating of $\mathcal{M}$.

Jointly optimizing parameters on $\mathcal{D}^t$ and $\mathcal{M}$ has several drawbacks [32, 4]. From a network capacity view, exhausting the parameter space early makes interference from the latter tasks more likely. On a generative stand, the deterministic mapping F^t limits capturing the randomness behind the real-world data. Owing to these, we next propose extending Eq. (2) with Eq. (1) to arrive at a CL model that: (a) allocates minimal parameters to learn robust per-task and global priors, and (b) uses generative factors to meet data-driven challenges such as deducing the right parameters for inference.

4.1 Neural Processes for Continual Learning

We begin with a direct extension of the NP formulation to CL. In an ER setup, where the context and target could be from t tasks, Eq. (1) can be extended to derive the joint posterior for NPs [13] as:

$$p(Y^{0:t}_{\mathcal{T}}|\mathcal{C}^{0:t}, X^{0:t}_{\mathcal{T}}) = \int p(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, z^G)p(z^G|\mathcal{C}^{0:t})dz^G, \tag{3}$$

where z^G models the joint distribution $F^{0:t}_*$ of CL tasks and is an enabler of the knowledge transfer [32] between these (see App. A.3 for ELBO). Eq. (3) still poses two challenges. First, it needs the labeled context $\mathcal{C}$ for inferring predictions, which is impossible in the CL setups where test data are assumed to be unlabeled. To overcome this, we turn to using the memory $\mathcal{M}$ offered by the ER-based setups as *context* during inference. Second, jointly modeling $F^{0:t}_*$ ignores the dynamics of per-task stochasticities, and is still prone to the bottlenecks of Eq. (2). Addressing the latter, we next consider *redefining* Eq. (3).

4.2 NPs with Hierarchical Task-specific Priors for CL

To learn informative task priors with knowledge transfer, we presume two solid directions. First, inducing the knowledge transfer implies that we preserve the global latent z^G. Second, the task priors could be captured better if modeled explicitly. We thus extend Eq. (3) with task-specific latents $z^t = (z^0, .., z^t)$. As a result, our posterior is a two-step hierarchical latent variable model (Fig. 2) where the global and the per-task latents model the inter and intra-task correlations, respectively:

$$p(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}^{0:t}) = \int\int \Big[\prod_{t=0}^{T-1} p(Y^t_{\mathcal{T}}|X^t_{\mathcal{T}}, z^t)p(z^t|z^G, \mathcal{C}^t)\Big]p(z^G|\mathcal{C}^{0:t})dz^{0:t}dz^G, \tag{4}$$

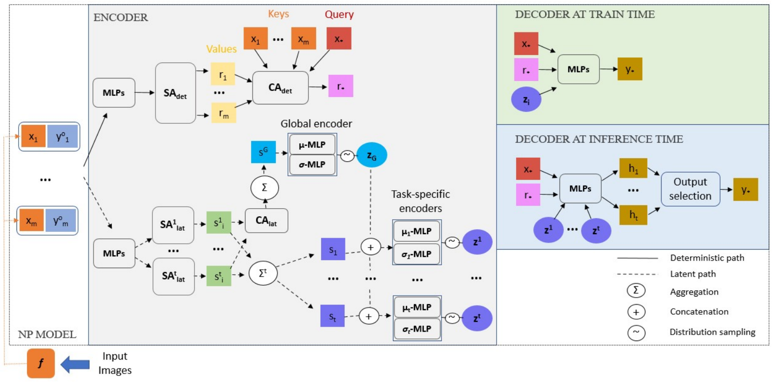


Figure 2: **NPCL architecture:** the decoding mechanism differs during training and inference.

where the entire context $\mathcal{C}^{0:t}$ is first encoded into z^G and then conditioned on z^G, the task-specific context $C^t := (C^0, .., C^t)$ are encoded into their respective latents. We refer to Eq. (4) as NP for CL (NPCL). NPCL generalizes to the MTP [23] when the input labels span the entire output spaces (see section 2). Next, we detail the neural network architecture for NPCL.

4.3 NPCL Architecture

Given the input images $x_i \in \{\mathcal{C}, \mathcal{T}\}$, we first pass these to a feature extractor f. With a slight abuse of notation, we denote the features as $x_i : x_i \in \mathbb{R}^{|f|}$ from here onward. x_i concatenated with the one-hot encoded labels $[x_i; y_i]$ is fed to the NPCL encoder with a deterministic and a latent path, and then to the decoder. All NPCL layers use multi-layer perceptrons (MLPs) projections, *i.e.,* $\text{MLP}(x) : \mathbb{R}^{|f|} \to \mathbb{R}^{|o|}$ where, o is a hyperparameter. We denote a normal distribution with a mean μ and a variance σ^2 by $\mathcal{N}(\mu, \sigma^2)$; the global and the task-specific distributions are $\mathcal{N}(\mu_G, \sigma_G^2)$ and $\mathcal{N}(\mu_t, \sigma_t^2)$. Lastly, by attention, we refer to the multi-head dot-product operations [43].

Latent Encoder. The latent path comprises of the projection $\Phi_i^{\text{lat}} = \text{MLP}([x_i; y_i])$ followed by two attention operations. First, per-task projections form the keys, values and queries to taskwise self-attention layers SA^t_{lat} that produce order-invariant encodings s_i^t over the samples of task t. Second, all encodings $\{s_i^{0:t}\}_{i=1}^{n+m}$ serve as the keys, values and queries to cross-attention layers $CA^{0:t}_{\text{lat}}$ that enrich their order-invariance from intra-task s^t to inter-task s^G. s^t and s^G are used to derive the N and M Monte Carlo samples of the global z^G and the task-specific latents z^t, respectively (see App. B for more details) using the reparameterization trick [26]. We set $M = 1$ to enhance the inter-task stochasticity in posterior. For each input, we thus get $N * (t + 1)$ latent outputs.

Deterministic Encoder. The deterministic path is similar to that of the ANP [24] and outputs an order-invariant representation r_* for target x_* (see App. B).

Decoder. Based on the task information, the decoder adopts separate mechanisms during training and inference. At train time, we use the available task labels to filter the N true latents $\{z_i^t\}_{i=1}^N$, combine them with r_* and x_*, and decode the logits h_*. We discuss the inference time decoding in sec. 4.5.

4.4 Learning Objectives for NPCL

The learning of NPCL is done by variational inference that maximizes the evidence lower bound with additional regularizations.

Evidence Lower Bound (ELBO). The intractability of Eq. (4) leads us to the following ELBO:

$$\begin{aligned}\log p_\theta(Y_{\mathcal{T}}^{0:t}|X_{\mathcal{T}}^{0:t}, \mathcal{C}) \geq \mathbb{E}_{q_\phi(z|\mathcal{T})}\Big[\sum_{t=0}^{T-1} \mathbb{E}_{q_\phi(z^t|z^G, \mathcal{C}^t)}[\log p_\theta(Y_{\mathcal{T}}^t|X_{\mathcal{T}}^t, z^t)] \\ - D^t\Big(q_\phi(z^t|z^G, \mathcal{T}^t) \| q_\phi(z^t|z^G, \mathcal{C}^t)\Big)\Big] - D^G\Big(q_\phi(z^G|\mathcal{T}) \| q_\phi(z^G|\mathcal{C})\Big),\end{aligned} \tag{5}$$

where $p_\theta(Y_\mathcal{T}^t|X_\mathcal{T}^t, z^t)$ is approximated by the CE loss. D^t and D^G denote the KL divergences between the approximate posterior and prior for the task-specific and global distributions, respectively. We derive the ELBO in App. A.1. We next identify two key aspects of forgetting in NPCL. In the following, we use D to denote the Jenshen-Shannon (JS) divergence [10] between two distributions.

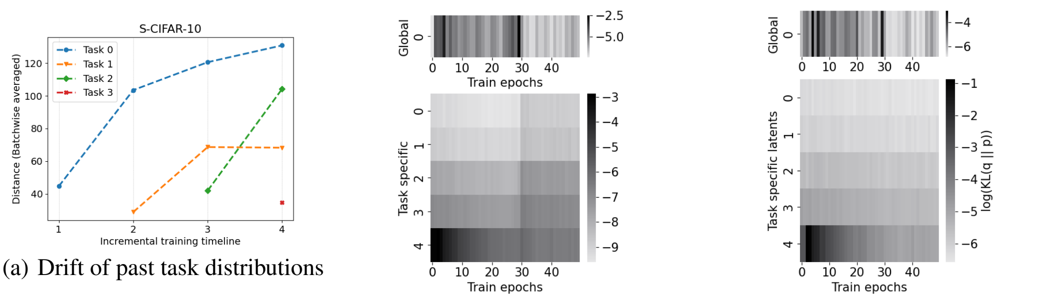
Global Regularization (GR). The training data of a CL task t is dominated by the t-*th* task samples. For NPCL, this drifts the global distribution $\mathcal{N}(\mu_G^t, \sigma_G^t)$ of past tasks towards the new task (Fig. 1). We thus regularize their global distribution using the one learned at step $t-1$:

$$\mathcal{L}_{\text{GR}} = D\big(\mathcal{N}(\mu_G, \sigma_G^2)_t, \mathcal{N}(\mu_G, \sigma_G^2)_{t-1}\big), \tag{6}$$

Task-specific Regularization (TR). While GR helps preserve the joint distribution of the past tasks, the hierarchy in NPCL leaves their task-specific distributions to be still prone to forgetting (Fig. 3(a)). This can further amplify the posterior collapse [41] for past task-specific latents during incremental training (Fig. 3(b)). To alleviate these, we regularize the learning of previous task distributions as:

$$\mathcal{L}_{\text{TR}}^t = D\big(\mathcal{N}(\mu_t, \sigma_t^2)_t, \mathcal{N}(\mu_t, \sigma_t^2)_j\big), \tag{7}$$

where j is the step at which the task t arrived. Given the reliance of Eq. (6) and Eq. (7) on past distributions, we maintain a separate buffer, which we refer to as the distribution memory $\mathcal{M}_\mathcal{N}$, to store the global $\mathcal{N}(\mu_G, \sigma_G^2)$ and the task-specific distributions $\mathcal{N}(\mu_{0:t-1}, \sigma_{0:t-1}^2)$. $\mathcal{M}_\mathcal{N}$ is updated after each incremental training step where we run an additional pass over the training data of task t alongside replaying $\mathcal{M}$ to record the batchwise averaged global and task-specific means and variances.



(a) Drift of past task distributions

(b) $\log KL(q|p)$ w/o TR

(c) $\log KL(q|p)$ w/ TR

Figure 3: **Need for distribution regularization:** Fig. 3(a) show the increasing distances between current distributions of past tasks and their original distributions (learned while the tasks were introduced); Fig. 3(b) and 3(c) show the effect of global (GR) and task regularization (TR) on the activation of the global and task-specific latent units. Low KL corresponds to an inactive unit.

Integrated Objective. Using α, β, γ and δ to denote the loss weights, our total loss can be given as:

$$\mathcal{L} = \frac{1}{|\mathcal{D}^t| + |\mathcal{M}|} \sum_{x,t \in \{\mathcal{D}^t \cup \mathcal{M}\}} (\mathcal{L}_{\text{CE}} + \alpha D^t + \beta D^G) + \frac{1}{|\mathcal{M}|} \sum_{x,t \in \mathcal{M}} \gamma \mathcal{L}_{\text{GR}} + \delta \mathcal{L}_{\text{TR}}^t, \tag{8}$$

where CE, D^t, and D^G act on the current task data $\mathcal{D}^t$ and on the buffer $\mathcal{M}$ while GR and TR act only on $\mathcal{M}$. By setting $0 < \{\alpha, \beta, \gamma, \delta\} < 1$, we resort to using the (respective) cold posteriors [49].

4.5 Inference with Uncertainty Awareness

NPCL's inference uses f to obtain the features x_* for the target test images. However, now we have no information on the target task label t. This leaves us with $\{z_i^{0:t}\}_{i=1}^N$ possible variables to infer our predictions from. A *naive* get around is to average over $N*(t+1)$ logits. But as the number of tasks grows, the noise from incorrect task priors would dominate the posterior. We thus propose using entropy as an uncertainty quantification metric (UQM) to filter the logits of the true task head ψ^t:

$$h_* = \arg\min_{j \in t} U(h_j), \quad U(h) = -\sum_{i \in N} \delta(i) \log(\delta(i)), \tag{9}$$

where δ is the $\mathsf{softmax}$ function and U is the total Shannon entropy [40] over the N logits per head. As we use true head ψ_ϕ^t during training, $\phi \in \psi^t$ produces low entropy for within distribution data.

Method	S-CIFAR-10 *Class-IL*		S-CIFAR-100 *Class-IL*		S-Tiny-ImageNet *Class-IL*		P-MNIST *Domain-IL*		R-MNIST *Domain-IL*	
Joint ResNet	92.2 $\pm$0.15		70.44		59.99$\pm$0.19		94.33$\pm$0.17		95.76$\pm$0.04	
Joint NP	91.66$\pm$0.11		70.58$\pm$0.24		59.83$\pm$0.17		95.02$\pm$0.21		95.37$\pm$0.07	
Joint ANP	91.26$\pm$0.16		70.77$\pm$0.21		60.14$\pm$0.17		95.39$\pm$0.18		95.85$\pm$0.05	
Joint NPCL	92.74$\pm$0.12		71.46$\pm$0.20		60.18$\pm$0.22		95.97$\pm$0.14		96.11$\pm$0.03	
Multitask NPCL	69.15$\pm$0.09		53.6$\pm$0.21		35.53$\pm$0.13		87.40$\pm$0.10		89.21$\pm$0.02	
oEWC [39]	19.49$\pm$0.12		-		7.58$\pm$0.10		75.79$\pm$2.25		77.35$\pm$5.77	
SI [48]	19.48 $\pm$0.17		-		6.58 $\pm$0.31		65.86$\pm$1.57		71.91 $\pm$5.83	
LwF [31]	19.61$\pm$0.05		-		8.46 $\pm$0.22		-		-	
$\mathcal{M}_{\text{size}}$	200	500	500	2000	200	500	200	500	200	500
ER [38]	44.79$\pm$1.86	57.74$\pm$0.27	22.10	38.58	8.49$\pm$0.16	9.99$\pm$0.29	72.37$\pm$0.87	80.6$\pm$0.86	85.01$\pm$1.90	88.91$\pm$1.44
iCaRL [37]	49.02 $\pm$3.20	47.55 $\pm$3.95	46.52	49.82	7.53$\pm$0.79	9.38$\pm$1.53	-	-	-	-
FDR [2]	30.91$\pm$2.74	28.71$\pm$3.23	-	-	8.70$\pm$0.19	10.54$\pm$0.21	74.77$\pm$0.83	83.18 $\pm$0.53	85.22$\pm$3.35	89.67$\pm$1.63
RPC [36]	-	-	22.34	38.33	-	-	-	-	-	-
DER [4]	61.93$\pm$1.79	70.51$\pm$1.67	36.6	51.89	11.87 $\pm$0.78	17.75$\pm$1.14	81.74$\pm$1.07	87.29$\pm$0.46	90.04$\pm$2.61	92.24$\pm$1.12
NP [13]	46.1$\pm$3.44	59.3$\pm$2.76	22.92	38.70	8.32$\pm$0.62	10.2$\pm$0.34	70.02$\pm$1.44	79.44$\pm$0.81	85.03$\pm$2.7	88.16$\pm$1.66
ANP [24]	46.67$\pm$1.23	58.77$\pm$0.65	23.2	39.06	8.81$\pm$0.93	9.75$\pm$0.90	73.55$\pm$0.66	80.98$\pm$0.57	85.70$\pm$1.39	89.21$\pm$0.93
ST-NPCL (w/ only per-task latent)	54.6$\pm$2.14	65.22$\pm$1.89	28.45	42.1	10.92$\pm$1.03	13.7$\pm$1.35	76.4$\pm$1.62	82.06$\pm$0.92	86.99$\pm$3.07	89.64$\pm$2.11
Naive NPCL (w/o task head inf.)	19.54$\pm$3.44	20.71$\pm$3.09	18.27	18.90	7.19$\pm$1.02	8.48$\pm$0.90	68.37$\pm$1.58	73.3$\pm$0.81	81.13$\pm$2.91	83.69$\pm$2.24
NPCL (ours)	63.78$\pm$1.70	71.34$\pm$1.48	37.43	46.71	12.44$\pm$0.59	15.29$\pm$1.02	83.11$\pm$0.90	86.52$\pm$0.77	91.48$\pm$1.79	92.07$\pm$1.39

Table 1: Classification accuracy for standard CL benchmarks across 10 runs. Best results are in red. Second best results are in blue. All runs of NP variants in the CL settings rely on ER. S-CIFAR-100 results are reported from Boschini et al. [3] while the rest are taken from Buzzega et al. [4].

5 Experiments

5.1 Settings

Datasets. We evaluate NPCL on class and domain incremental learning (IL) settings. For class-IL, we use three public datasets: sequential CIFAR10 (S-CIFAR-10) [32], sequential CIFAR100 (S-CIFAR-100) [48], and sequential Tiny ImageNet (S-Tiny-ImageNet) [6]. For domain-IL, we use Permuted MNIST (P-MNIST) [27] and Rotated MNIST (R-MNIST) [32]. S-CIFAR-10, S-CIFAR-100, and S-Tiny-ImageNet host 10, 100, and 200 classes each with 5000, 500, and 500 training images and 1000, 100, and 50 test images per class, respectively. The number of sequential tasks for S-CIFAR-10 is 5 (2 classes per task); for S-CIFAR-100 and S-Tiny-ImageNet is 10 (10 and 20 classes per task, respectively); for P/R-MNIST is 20 where P-MNIST creates tasks out of MNIST [29] by randomly permuting the image pixels while R-MNIST does so by rotating images randomly in the range $[0, \pi)$.

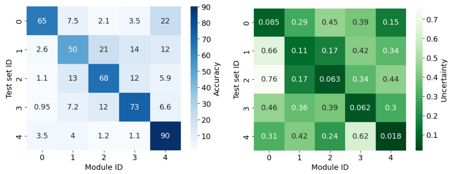
Architectures. For a fair comparison against other methods, we rely on the Mammoth CL benchmark [3]. Our backbone for class-IL experiments is a ResNet-18 [19] without pretraining, while for domain-IL, we rely on a fully connected (FC) network with two hidden layers [32]. NPCL relies on Xavier initialized [14] FC layers with: two 256-d hidden layers for class-IL and one 32-d layer for domain-IL setups. For class-IL, each FC layer is followed by layer normalization [1] and ReLU.

Configuration and Hyperparameters. We train all models using SGD optimizer. The number of training epochs per task for S-Tiny-ImageNet is 100, for S-CIFAR-(10/100) is 50 and that for (P/R)-MNIST is 1. We detail further on configurations, hyperparameters, and their tuning in App. C.

Baselines. We employ several CL methods to compare NPCL with. Regularization-based methods include oEWC [39] and SI [48]; knowledge distillation-based methods include iCaRL [37] and LwF [31]; rehearsal-based methods are ER [38], RPC [36], FDR [2], DER [4]. Among neural processes, we use NP [13], ANP [24] with only global latent and Single Task (ST) NPCL (see App. A.2) with only per-task latents. We use five non-CL benchmarks for upper bounds on the performance: Joint ResNet / NP / ANP / NPCL perform joint training of all tasks using a single task head while the multitask NPCL infers task heads in joint training using Eq. (9). Finally, Naive NPCL infers predictions by averaging logits of all task heads.

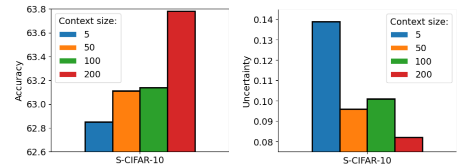
5.2 Results

Table 1 reports the average accuracy after training on all tasks. Across all settings, NPCL boosts the performance of ER and achieves either comparable results compared with the state-of-the-art (SOTA), *e.g.*, DER. Compared to regularization-based oEWC and SI, NPCL obtains a significant gain



(a) Accuracy (b) Uncertainty

Figure 4: Heatmaps depicting the taskwise averaged accuracy and uncertainty of test samples per task head on S-CIFAR-10.



(a) Accuracy (b) Uncertainty

Figure 5: Effect of context set size ($|\mathcal{M}| = \{5, 50, 100, 200\}$) on the accuracy and uncertainty of NPCL on S-CIFAR-10.

in performance. This is because the former methods calculate weight importance which is liable to changes with new tasks. Regularizing explicitly towards the global and per-task distributions of past tasks helps NPCL overcome this. Further, on both class and domain-IL, NPCL stands out in the most challenging setting where the episodic memory size is the smallest. On domain-IL where the shift occurs within the domain instead of classes, the performance of a number of methods degrade as they forget the relations among a task's classes. Preserving the tasks' distributions helps NPCL maintain valuable information in this case. Analyzing the backward transfer (BWT) scores [34] shows that NPCL's forgetting is competitive or lesser than the SOTA (see Table 8). Lastly, we observe that ST-NPCL with no hierarchy lags in BWT and accuracy due to limited knowledge transfer between tasks.

5.3 Ablation Studies

On Learning Objectives. Table 2 shows the impact of distribution regularization with the baseline being an NPCL trained with no regularization. We observe that the baseline performs worse than the ResNet-based ER as the NPCL layers are liable to more forgetting. Including TR into our objectives leads to the singlemost gain over the baseline. We further study how these objectives guide the learning of the global and task-specific means and variances with training (see App. E.1). We observe that NPCL w/ TR leads to better learning of the current task as well as preserving the past task distributions but at the cost of drifting the global distribution. NPCL w/ GR restricts the global distribution drift but not for the per-task distributions. NPCL w/ GR and TR strikes a balance in between.

Method	S-CIFAR-10	S-Tiny-ImageNet
ER	44.79	8.49
Baseline (w/o GR or TR)	32.24	7.15
NPCL (w/ only GR)	50.68	8.61
NPCL (w/ only TR)	57.28	11.36
NPCL (w/ GR and TR)	**63.78**	**12.44**

Table 2: Accuracy w/ learning objectives

On Uncertainty. Fig. 4 ablates the average accuracies and uncertainties of each task head predictions over the test set of each task on S-CIFAR-10 (see App. E.2 for S-CIFAR-100). First, we observe that the accuracy of predictions made by true task heads are, in general, a magnitude higher than the rest. For uncertainty, this trend is reversed. This verifies our assumption that restricting latent heads to learn only their true label distribution makes them more confident in modeling the within-task samples. Second, for recently trained tasks, the uncertainty differences between the true task heads and the rest are greater than the earlier tasks. This suggests that the extent of forgetting goes beyond a model's accuracy and to other aspects of learning such as its confidence. To further verify this, we probe the BWT of uncertainty, and see a strong correlation with the BWT of accuracy (see Fig. 7).

On Context Size. We study the average accuracy (Fig. 5(a)) and uncertainty (Fig. 5(b)) after training on S-CIFAR-10 with $|\mathcal{M}| = 200$, and then varying the context sizes during inference. Similar to other NPs [44, 11], we find a positive correlation between context size and performance indicating that NPCL utilizes useful information from diverse context, thereby reducing its task inference ambiguity.

On Storage Overhead. For each task, NPCL stores two new vectors – task-specific mean and variance, and replaces the global mean and variance with the current global ones. The NPCL storage thus scales constantly in the size $|\mathcal{M}|$ of the memory. This offers a strong edge on storage efficiency when compared to the SOTA [4] scaling quadratically, i.e., $|\mathcal{M} * N_C|$ where N_C is the total number of classes. For instance, on S-Tiny-ImageNet with $|\mathcal{M}| = 500, |N_C| = 200$, NPCL's cumulative storage amounts to a (flattened) vector of size 6132 (256*10*2 for 256-d means and variances of 10 tasks + 256*2 for 256-d global mean and variance + 500 for 1-d task labels) while that of DER amounts to 100,000 (200*500 for logits of 500 memory samples), *i.e.,* **a 93.868% storage gain**. We report the storage gains of NPCL over DER across all settings in App. E.3.

5.4 Applications of Uncertainty Quantification

The probabilistic nature of NPCL offers it an edge at leveraging data-driven UQMs. To further study the usage its predictive uncertainties, we design two experiments that leverage pretrained NPCL.

Novel Data Identification. Novel data identification seeks to distinguish out-of-distribution data ($\mathcal{D}_{\text{OOD}}$) from in-domain data ($\mathcal{D}_{\text{ID}}$). Forgetting makes CL models struggle further on the task [18]. The probabilistic sampling in NPCL opens the door for leveraging its predictive variances – which are more reliable estimates of aleatoric uncertainty than pointwise predictions [21]. For the N predicted logits, we thus compute the variances over their softmax scores, $\sigma^2(\delta(h_*))$, and their uncertainty scores, $\sigma^2(U(h_*))$. Table 3 evaluates these metrics for ID (S-CIFAR-10) and OOD (first 10 classes of S-CIFAR-100) data after each task. We observe that the variance scores of either metrics on $\mathcal{D}_{\text{ID}}$ are up to a magnitude lower than those on $\mathcal{D}_{\text{OOD}}$. We further observe an overall decrease in the variances with the arrival of further incremental tasks. This could be attributed to the generalization of more low-level features in the novel data as in-domain [17, 15]. We detail further novel data identification experiments in App. E.4.

Incremental step	$\mathcal{D}_{\text{ID}}$ = CIFAR-10, $\mathcal{D}_{\text{OOD}}$ = CIFAR-100			
	$\mathcal{D}_{\text{ID}}$ (δ)	$\mathcal{D}_{\text{OOD}}$ (δ)	$\mathcal{D}_{\text{ID}}$ (H)	$\mathcal{D}_{\text{OOD}}$ (H)
1	$1e^{-6}$	$1e^{-5}$	$9.3e^{-6}$	$8.4e^{-5}$
2	$2.6e^{-6}$	$1.4e^{-5}$	$6.3e^{-5}$	$2.2e^{-4}$
3	$2.3e^{-6}$	$6.2e^{-6}$	$6.7e^{-5}$	$2.1e^{-4}$
4	$8.1e^{-7}$	$4.8e^{-6}$	$4.6e^{-5}$	$2.2e^{-4}$
5	$7.1e^{-7}$	$1.7e^{-6}$	$4.6e^{-5}$	$1.1e^{-4}$

Table 3: Average variances over softmax (δ) and entropy (H) scores on in- and out-of-domain test sets using $N = 50$ ancestral samples.

Instance-level Model Confidence Evaluation. The confidence evaluation framework of Han et al. [16] provides finer granularity for assessing the predictive confidence of classification models (see App. E.5 for more details and normality test). Table 4 shows the results of one run of the framework after training on S-CIFAR-10 (see App. Table 11 for all classes). Here, we use the task identity to select the latent head per class. We observe the mean prediction interval width (PIW) of the true class label among the correct predictions to be narrower than that of the incorrect predictions, implying that the NPCL's variations of predicted class labels is smaller when the predictions are correct. We also notice a higher accuracy among the test instances rejected by the *t*-test than those not rejected.

Class	Accuracy	PIW		Accuracy by t-test status	
		Correct	Incorrect	Rejected	Not Rejected
1	82.30	74.17	102.21	83.37	50.00
2	94.00	62.90	79.86	94.07	80.00
3	74.00	54.92	68.48	74.14	64.29

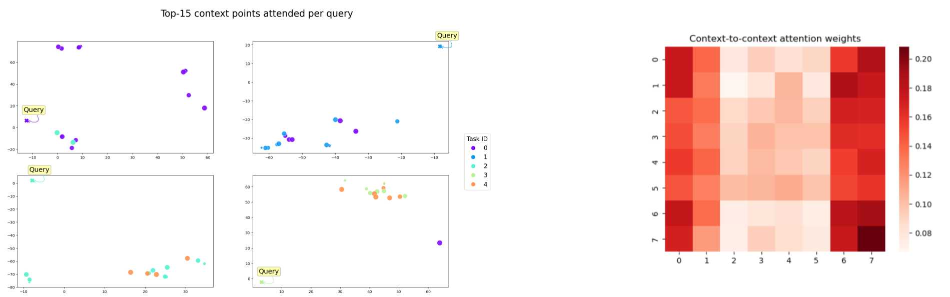
Table 4: PIW (multiplied by 100) and $t-$test results for the first three classes of S-CIFAR-10 inferred from their respective task heads.

6 Limitations

We list the key limitations of NPCL to facilitate future research directions. These include:

Incompetence of dot-product attention: Similar to the ANP [24], NPCL employs the permutation-invariant scaled-dot product attention [43] to weigh the relevant context and target embeddings. Visualizing the attention weights computed by the cross-attention layers of the deterministic path shows us that the top attended context for the target queries often contain points belonging to other CL tasks (Fig. 6(a)). This *limits* the performance sensitivity of NPCL with respect to the increase in context thus resulting in a lag of accuracy behind SOTA on CL setups with larger episodic memory sizes (see Table 1). To further verify the relevance of the attended context, we visualize the self-attention weights of all context points. Fig. 6(b) shows that the lowest or the maximum values in the context dataset have larger weights. Such an observation is in line with existing works pointing that the scaled-dot product attention can derive irrelevant set encodings of the context points and can thus lag at exploiting the context embeddings properly [25].

Computational overhead: Table 5 compares the number of parameters of the NPCL with ER / DER [3] where the latter rely solely on the ResNet-18 backbone as they do not exploit parameter isolation



(a) Cross-attention visualization

(b) Self-attention visualization

Figure 6: **Scaled dot-product attention visualization**: (a) top-15 context (buffer) points attended for 4 randomly chosen queries (test set samples). The queries are made after training on S-CIFAR-10. The sizes of the points correspond to the attention values while the colors denote the tasks they belong to. (b) self-attention weights of context points when all feature values are arranged in ascending order shows that ANP [24] mostly attends to the lowest or the maximum values in the context dataset.

for task heads. Overall, the percentage increase in parameter number is 57.6% for S-CIFAR-10, 46.57% for S-CIFAR-100 and S-Tiny-ImageNet, and 55.25% for P/R-MNIST.

Method / Dataset	**S-CIFAR-10**	**S-CIFAR-100**	**S-Tiny-ImageNet**	**P/R-MNIST**
ER / DER [1]	11,173,962	11,220,132	11,220,132	89,610
NPCL	19,397,706	24,091,556	24,091,556	162,166

Table 5: Comparison of the total number of parameters for ER / DER against NPCL.

Inference time complexity: The reliance on self-attention means that the inference time complexity of NPCL is $\mathcal{O}(n * m)$ where, n is the number of context points (sampled from the episodic memory) and m is the number of target points (the number of test samples). Due to this, the runtime for inference scales polynomially with the number of context points (sampled from the buffer). Table 6 reports the runtime of NPCL on S-CIFAR-10 and S-CIFAR-100 settings by varying the context sizes. For reference, the first row reports the runtime of ER / DER whose inference complexity is O(1) in the memory buffer size.

Method	**S-CIFAR-10**
ER / DER	3.72s
NPCL,$\lvert\mathcal{M}\rvert = 200$	19.58s
NPCL, $\lvert\mathcal{M}\rvert = 500$	31.25s
NPCL, $\lvert\mathcal{M}\rvert = 1000$	47.99s
NPCL, $\lvert\mathcal{M}\rvert = 2000$	84.86s

Table 6: Inference time with varying context sizes

Incompatibility with logits-based replay: NPCL is incompatible with logits based replay because of the stochasticity in the posterior induced by Monte Carlo sampling. Overcoming this could help boost the performance of NPCL further over SOTA like DER [4] and DER++ [3].

7 Conclusion

In this paper, we propose neural processes for continual learning (NPCL), a hierarchical latent variable setup designed to jointly model the task-agnostic and task-specific data generating functions in CL. We study the potential forgetting aspects in NPCL and propose to regularize the previously learned distributions at a global and a per-task granularity. We further demonstrate that using entropy as an uncertainty quantification metric (UQM) helps NPCL infer correct task heads and boost the performance of baseline experience replay to even surpass state-of-the-art deterministic models on several CL settings. We further show out-of-the-box applications of the uncertainty estimation capabilities of NPCL for novel data identification and instance-level confidence evaluation. We conclude our ablations by listing the key limitations of NPCL, which we hope could lay solid directions for further research on uncertainty-aware CL.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. 2016.

[2] Ari Benjamin, David Rolnick, and Konrad Kording. Measuring and regularizing networks in function space. In *International Conference on Learning Representations*, 2018.

[3] Matteo Boschini, Lorenzo Bonicelli, Pietro Buzzega, Angelo Porrello, and Simone Calderara. Class-incremental continual learning into the extended der-verse. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[4] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and SIMONE CALDERARA. Dark experience for general continual learning: a strong, simple baseline. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15920–15930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/b704ea2c39778f07c617f6b7ce480e9e-Paper.pdf.

[5] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-GEM. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Hkf2_sC5FX.

[6] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

[7] Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems*, 33: 9900–9911, 2020.

[8] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9285–9295, 2022.

[9] Xinjie Fan, Shujian Zhang, Korawat Tanwisuth, Xiaoning Qian, and Mingyuan Zhou. Contextual dropout: An efficient sample-dependent dropout module. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=ct8_a9h1M.

[10] Bent Fuglede and Flemming Topsoe. Jensen-shannon divergence and hilbert space embedding. In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, page 31. IEEE, 2004.

[11] Ning Gao, Hanna Ziesche, Ngo Anh Vien, Michael Volpp, and Gerhard Neumann. What matters for meta-learning vision regression tasks? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14776–14786, 2022.

[12] Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *International Conference on Machine Learning*, pages 1704–1713. PMLR, 2018.

[13] Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

[14] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[15] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

[16] Xizewen Han, Huangjie Zheng, and Mingyuan Zhou. CARD: Classification and regression diffusion models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=4L2zYEJ9d_.

[17] Jakob D Havtorn, Jes Frellsen, Søren Hauberg, and Lars Maaløe. Hierarchical vaes know what they don't know. In *International Conference on Machine Learning*, pages 4117–4128. PMLR, 2021.

[18] Jiangpeng He and Fengqing Zhu. Out-of-distribution detection in unsupervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3850–3855, 2022.

[19] 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*, pages 770–778, 2016.

[20] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=Hkg4TI9xl`.

[21] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110:457–506, 2021.

[22] Saurav Jha, Dong Gong, Xuesong Wang, Richard E Turner, and Lina Yao. The neural process family: Survey, applications and perspectives. *arXiv preprint arXiv:2209.00517*, 2022.

[23] Donggyun Kim, Seongwoong Cho, Wonkwang Lee, and Seunghoon Hong. Multi-task processes. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=9otKVlgrpZG`.

[24] Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=SkE6PjC9KX`.

[25] Mingyu Kim, Kyeong Ryeol Go, and Se-Young Yun. Neural processes with stochastic attention: Paying more attention to the context dataset. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=JPkQwEdYn8`.

[26] Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 28, 2015.

[27] 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.

[28] Yann LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62, 2022.

[29] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[30] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, pages 3925–3934. PMLR, 2019.

[31] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[32] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

[33] Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects, 2013.

[34] Guy Oren and Lior Wolf. In defense of the learning without forgetting for task incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2209–2218, 2021.

[35] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.

[36] Federico Pernici, Matteo Bruni, Claudio Baecchi, Francesco Turchini, and Alberto Del Bimbo. Class-incremental learning with pre-allocated fixed classifiers. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6259–6266. IEEE, 2021.

[37] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *ProceedinFgs of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[38] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, , and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=B1gTShAct7`.

[39] 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*, pages 4528–4537. PMLR, 2018.

[40] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[41] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. *Advances in neural information processing systems*, 29, 2016.

[42] Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, pages 1–13, 2022.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[44] Jianfeng Wang, Thomas Lukasiewicz, Daniela Massiceti, Xiaolin Hu, Vladimir Pavlovic, and Alexandros Neophytou. Np-match: When neural processes meet semi-supervised learning. In *International Conference on Machine Learning*, pages 22919–22934. PMLR, 2022.

[45] Qi Wang and Herke Van Hoof. Doubly stochastic variational inference for neural processes with hierarchical latent variables. In *International Conference on Machine Learning*, pages 10018–10028. PMLR, 2020.

[46] Qingsen Yan, Dong Gong, Yuhang Liu, Anton van den Hengel, and Javen Qinfeng Shi. Learning bayesian sparse networks with full experience replay for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 109–118, 2022.

[47] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Sk7KsfW0-`.

[48] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

[49] Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy natural gradient as variational inference. In *International conference on machine learning*, pages 5852–5861. PMLR, 2018.

A Theory

A.1 ELBO derivation for NPCL

Borrowing the conventions from section 3, for an incremental task $0 \leq t \leq T$, we assume the context $\mathcal{C}$ and targets $\mathcal{T}$ to comprise of samples from all t seen classes. Accordingly, we define these as $\mathcal{C} = (X^{0:t}_{\mathcal{C}}, Y^{0:t}_{\mathcal{C}})$ and $\mathcal{T} = (X^{0:t}_{\mathcal{T}}, Y^{0:t}_{\mathcal{T}})$, respectively. To enforce the prior that both $\mathcal{C}$ and $\mathcal{T}$ follow the same distribution, we assume $\mathcal{C}^t \subset \mathcal{T}^t$, and therefore, $\mathcal{C} \subset \mathcal{T}$. In order to derive predictions $Y^{0:t}_{\mathcal{T}}$ on $X^{0:t}_{\mathcal{T}}$, the NPCL relies on the context $\mathcal{C}$ to build conditional priors $p_\theta(z^G|\mathcal{C})$ and $p_\theta(z^t|z^G, \mathcal{C}^t)$, where p_θ is the decoder. The decoder's objective thus boils down to maximizing the log-likelihood of the observations, *i.e.,* the evidence $\log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C})$. In what follows, we derive the evidence lower bound (ELBO):

$$
\begin{aligned}
&\log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}) && \text{(Log-likelihood of evidence)} \quad (10a)\\
&= \log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}) \int p_\theta(z^G|X^{0:t}_{\mathcal{T}}, Y^{0:t}_{\mathcal{T}}, \mathcal{C}) dz^G && \Big(\because \int p_\theta(z^G|\mathcal{T}, \mathcal{C}) dz^G = 1\Big) \quad (10b)\\
&= \int p_\theta(z^G|X^{0:t}_{\mathcal{T}}, Y^{0:t}_{\mathcal{T}}, \mathcal{C}) (\log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C})) dz^G && \text{(Integrate over the log-likelihood)} \quad (10c)\\
&= \mathbb{E}_{q_\phi(z^G|\mathcal{T})}[\log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C})] && \text{(By definition)} \quad (10d)\\
&= \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log \frac{p_\theta(Y^{0:t}_{\mathcal{T}}, z^G|X^{0:t}_{\mathcal{T}}, \mathcal{C})}{p_\theta(z^G|X^{0:t}_{\mathcal{T}}, Y^{0:t}_{\mathcal{T}}, \mathcal{C})}\Big] && \text{(Re-introduce } z^G \text{ by Chain rule)} \quad (10e)\\
&= \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log \frac{p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}, z^G) p_\theta(z^G|X^{0:t}_{\mathcal{T}}, \mathcal{C})}{p_\theta(z^G|\mathcal{T})}\Big] && \text{(Chain rule of probability; } \mathcal{C} \subset \mathcal{T}) \quad (10f)\\
&= \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log \frac{p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}, z^G) p_\theta(z^G|X^{0:t}_{\mathcal{T}}, \mathcal{C}) q_\phi(z^G|\mathcal{T})}{p_\theta(z^G|\mathcal{T}) q_\phi(z^G|\mathcal{T})}\Big] && \text{(Equivalent fraction)} \quad (10g)\\
&= \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}, z^G)\Big] \\
&\qquad + \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log \frac{p_\theta(z^G|\mathcal{C})}{q_\phi(z^G|\mathcal{T})}\Big] + \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log \frac{q_\phi(z^G|\mathcal{T})}{p_\theta(z^G|\mathcal{T})}\Big] && \text{(Split the expectation)} \quad (10h)\\
&= \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}, z^G)\Big] \\
&\qquad - D_{\mathrm{KL}}\big(q_\phi(z^G|\mathcal{T}) \| p_\theta(z^G|\mathcal{C})\big) + D_{\mathrm{KL}}\big(q_\phi(z^G|\mathcal{T}) \| p_\theta(z^G|\mathcal{T})\big) && \text{(By definition of KL divergence)} \quad (10i)\\
&\geq \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}, z^G)\Big] - D_{\mathrm{KL}}\big(q_\phi(z^G|\mathcal{T}) \| p_\theta(z^G|\mathcal{C})\big) && (\because \text{KL divergence for posterior approximation } \geq 0) \quad (10j)
\end{aligned}
$$

where the evidence is equal to the sum of the reconstruction likelihood $\mathbb{E}_{q_\phi(z^G|\mathcal{T})}\big[\log p_\theta(Y^{0:t}_{\mathcal{T}}|X^{0:t}_{\mathcal{T}}, \mathcal{C}, z^G)\big]$ of the decoder and the KL divergence between the true posterior $p_\theta(z^G|\mathcal{T})$ and the approximate posterior $q_\phi(z^G|\mathcal{T})$ learned using the variational distribution, minus the prior matching term $D_{\mathrm{KL}}\big(q_\phi(z^G|\mathcal{T}) \| p_\theta(z^G|\mathcal{C})\big)$. In particular, NPCL learns two approximate distributions $q_\phi(z^G|\mathcal{T})$ and $q_\phi(z^t|z^G, \mathcal{T}^t)$, that seek to estimate the global posterior $p_\theta(z^G|\mathcal{T})$ and the task-specific posterior $p_\theta(z^t|z^G, \mathcal{T}^t)$. To realize the latter posterior, we introduce the hierarchy of task-specific latents $z^{0:t}$. This allows us to expand and derive a lower bound to the reconstruction likelihood as:

$$\log p_\theta(Y_{\mathcal{T}}^{0:t}|X_{\mathcal{T}}^{0:t},\mathcal{C},z^G) \quad \text{(Reconstruction term)} \tag{11a}$$

$$= \mathbb{E}_{\prod_0^t q_\phi(z^t|z^G,\mathcal{T}^t)}\Big[\log \frac{p_\theta(Y_{\mathcal{T}}^{0:t},z^{0:t}|X^{0:t},\mathcal{C},z^G)}{p_\theta(z^{0:t}|X_{\mathcal{T}}^{0:t},Y_{\mathcal{T}}^{0:t},\mathcal{C},z^G)}\Big] \quad \text{(Introduce one-level latent hierarchy)} \tag{11b}$$

$$= \mathbb{E}_{\prod_0^t q_\phi(z^t|z^G,\mathcal{T}^t)}\Big[\log \int_0^t \frac{p_\theta(Y_{\mathcal{T}}^t,z^t|X^t,\mathcal{C}^t,z^G)}{p_\theta(z^t|X_{\mathcal{T}}^t,Y_{\mathcal{T}}^t,\mathcal{C}^t,z^G)}\Big] \quad \text{(Integrate over individual tasks)} \tag{11c}$$

$$= \int_0^t \mathbb{E}_{q_\phi(z^t|z^G,\mathcal{T}^t)}\Big[\log \frac{p_\theta(Y_{\mathcal{T}}^t|X_{\mathcal{T}}^t,\mathcal{C}^t,z^G,z^t)p_\theta(z^t|X_{\mathcal{T}}^t,\mathcal{C}^t,z^G)}{p_\theta(z^t|\mathcal{T}^t,z^G)}\Big] \quad \text{(Chain rule of probability; } \mathcal{C}\subset\mathcal{T}) \tag{11d}$$

$$= \int_0^t \mathbb{E}_{q_\phi(z^t|z^G,\mathcal{T}^t)}\Big[\log \frac{p_\theta(Y_{\mathcal{T}}^t|X_{\mathcal{T}}^t,z^t)p_\theta(z^t|\mathcal{C}^t,z^G)q_\phi(z^t|z^G,\mathcal{T}^t)}{p_\theta(z^t|\mathcal{T}^t,z^G)q_\phi(z^t|z^G,\mathcal{T}^t)}\Big] \quad \text{(Equivalent fraction)} \tag{11e}$$

$$\begin{aligned}= \int_0^t \mathbb{E}_{q_\phi(z^t|z^G,\mathcal{T}^t)}&\Big[\log p_\theta(Y_{\mathcal{T}}^t|X_{\mathcal{T}}^t,z^t)\Big] \\ &- D_{\text{KL}}\big(q_\phi(z^t|z^G,\mathcal{T}^t)\|p_\theta(z^t|z^G,\mathcal{C}^t)\big) + D_{\text{KL}}\big(q_\phi(z^t|z^G,\mathcal{T}^t)\|p_\theta(z^t|z^G,\mathcal{T}^t)\big) \quad \text{(By definition of KL divergence)}\end{aligned} \tag{11f}$$

$$\geq \int_0^t \mathbb{E}_{q_\phi(z^t|z^G,\mathcal{T}^t)}\Big[\log p_\theta(Y_{\mathcal{T}}^t|X_{\mathcal{T}}^t,z^t)\Big] - D_{\text{KL}}\big(q_\phi(z^t|z^G,\mathcal{T}^t)\|p_\theta(z^t|z^G,\mathcal{C}^t)\big) \quad (\because \text{ KL divergence} \geq 0) \tag{11g}$$

Plugging Eq. (11g) into Eq. (10j), we get the final ELBO:

$$\log p_\theta(Y_{\mathcal{T}}^{0:t}|X_{\mathcal{T}}^{0:t},\mathcal{C}) \tag{12a}$$

$$\begin{aligned}\geq \mathbb{E}_{q_\phi(z^G|\mathcal{T})}&\Big[\int_0^t \mathbb{E}_{q_\phi(z^t|z^G,\mathcal{T}^t)}\Big[\log p_\theta(Y_{\mathcal{T}}^t|X_{\mathcal{T}}^t,z^t)\Big] - D_{\text{KL}}\big(q_\phi(z^t|z^G,\mathcal{T}^t)\|p_\theta(z^t|z^G,\mathcal{C}^t)\big)\Big] \\ &- D_{\text{KL}}\big(q_\phi(z^G|\mathcal{T})\|p_\theta(z^G|\mathcal{C})\big) \quad \text{(By substitution)}\end{aligned} \tag{12b}$$

$$\begin{aligned}= \mathbb{E}_{q_\phi(z^G|\mathcal{T})}&\Big[\int_0^t \mathbb{E}_{q_\phi(z^t|z^G,\mathcal{T}^t)}\Big[\log p_\theta(Y_{\mathcal{T}}^t|X_{\mathcal{T}}^t,z^t)\Big] - D_{\text{KL}}\big(q_\phi(z^t|z^G,\mathcal{T}^t)\|q_\phi(z^t|z^G,\mathcal{C}^t)\big)\Big] \\ &- D_{\text{KL}}\big(q_\phi(z^G|\mathcal{T})\|q_\phi(z^G|\mathcal{C})\big), \quad \text{(Final ELBO)}\end{aligned} \tag{12c}$$

where the decoder p_θ serves as the conditional prior network and is replaced by the encoder q_ϕ serving as the surrogate posterior network. q_ϕ can be seen to be producing two intermediate bottleneck distributions: (a) $q_\phi(z^G|\mathcal{T})$ transforms inputs into a distribution over global latents, (b) conditioned on the global latents, $q_\phi(z^t|z^G,\mathcal{T}^t)$ gathers the t-*th* task inputs and learns another distribution over the task-specific latents. The task-specific latents and their corresponding input covariates X^t are then used by the deterministic decoder p_θ to decode their corresponding logit h_*. It is indeed this dependency of p_θ on the task identifier t that makes inference a challenging task in real-world CL settings.

A.2 Single Task NPCL and its ELBO

Single Task (ST) NPCL preserves all but the inter-task cross attention $CA_{\text{lat}}^{0:t}$ and the global distribution encoder ψ^G layers from the architecture of NPCL (section 4.3). The task-specific latents z^t are thus derived as:

$$\{z_i^t\}_{i=1}^M \sim \mathcal{N}(\psi_\mu^t(s_i^t),\psi_\sigma^t(s_i^t)) = \mathcal{N}(\mu_t,\sigma_t^2), \forall t\in T, \tag{13}$$

where s_i^t and ψ^t carry the same meaning as in Eq. (16). For a fair comparison in Table 1, we fix M to be the same as the total number of global ancestral samples N in NPCL. The corresponding ELBO amounts to:

$$\log p_\theta(Y_{\mathcal{T}}^{0:t}|X_{\mathcal{T}}^{0:t},\mathcal{C}) \tag{14a}$$

$$\geq \int_0^t \mathbb{E}_{q_\phi(z^t|\mathcal{T}^t)}\Big[\log p_\theta(Y_{\mathcal{T}}^{t}|X_{\mathcal{T}}^{t},z^t)\Big] - D_{\mathrm{KL}}\big(q_\phi(z^t|\mathcal{T}^t)\|p_\theta(z^t|\mathcal{C}^t)\big) \quad \text{(Dropping } z^G \text{ from Eq. (12c))} \tag{14b}$$

A.3 ELBO for NP and ANP

NP [13] and ANP [24] employ a single latent variable z^G to model the global correlation of all tasks. In particular, compared to section 4.3, the task-specific self-attention layer SA^t_{lat} and the task-specific distribution encoder ψ^t is no longer required. While this enables knowledge sharing among tasks, NPs and ANPs are limited in modeling finer intra-task stochastic factors. The ELBO can be given as:

$$\log p_\theta(Y_{\mathcal{T}}^{0:t}|X_{\mathcal{T}},\mathcal{C}) \tag{15a}$$

$$\geq \mathbb{E}_{q_\phi(z^G|\mathcal{T})}\Big[\log p_\theta(Y_{\mathcal{T}}^{0:t}|X_{\mathcal{T}},z^G)\Big] - D_{\mathrm{KL}}\big(q_\phi(z^G|\mathcal{T})\|p_\theta(z^G|\mathcal{C})\big) \quad \text{(Dropping } z^{0:t} \text{ from Eq. (12c))} \tag{15b}$$

where z^G is derived in a way similar to Eq. (16), and the inputs $X_{\mathcal{T}}$ and $\mathcal{C}$ belong to $[0,t]$ tasks without relying on the task labels for being encoded.

B Further on NPCL Architecture

In the following, we denote multi-head dot product self-attention [43] by $SA(K,V,Q)$ where K, V, and Q are the keys, values and queries respectively. The equivalent notation for cross-attention is $CA(K,V,Q)$.

Latent Encoder. The latent path learns the functional prior and posterior from the context and the target sets, respectively. Each label-concatenated input is projected as $\Phi_i^{\text{lat}} = \text{MLP}([x_i;y_i])$; then subjected to two attention operations. First, per-task projections form the keys, values and queries to taskwise self-attention layers $SA^t_{\text{lat}}(\Phi_i^{\text{lat}},\Phi_i^{\text{lat}},\Phi_i^{\text{lat}}) : \Phi_i^{\text{lat}} \to s_i^t$ that produce order-invariant encodings s_i^t over the task t. Second, all encodings $\{s_i^{0:t}\}_{i=1}^{n+m}$ serve as the keys, values and queries to the cross-attention layers $CA^{0:t}_{\text{lat}}(s_i^t,s_i^t,s_i^t) : s_i^t \to s_i^G$ that enrich their order-invariance from intra-task s^t to inter-task s^G. s^t and s^G are then used to derive the global z^G and the task-specific latents z^t:

Such globally attended inputs are passed in parallel to two MLP layers constituting the global distribution encoder ψ^G whose outputs together parameterize the global distribution $\mathcal{N}(\mu_G,\sigma_G^2)$ over the input set, *i.e.,* $\psi^G(s^G) : \{s_i^G\}_{i=1}^{n+m} \to (\mu_G,\sigma_G^2)$. Samples $\{z_i^G\}_{i=1}^N$ drawn from this distribution are proxies for the variables capturing the global correlation over all tasks in the input set. It is indeed this sampling step that induces the stochasticity into the learned posteriors of the NPCL.

To model finer task-specific distribution for task t conditioned on the global distribution, we retain the task-specific self-attended representations s_i^t and concatenate these with the global latents $\{z_i^G\}_{i=1}^N$ to produce N distinct encodings per input point. These encodings are then passed through the t-*th* task distribution encoder ψ^t that again constitutes a mean and a variance MLP head and produces outputs that parameterize the t-*th* task distribution $\mathcal{N}(\mu_t,\sigma_t^2)$, *i.e.,* $\psi^t(s^t) : \{s_i^t\}_{i=1}^{n+m} \to (\mu_t,\sigma_t^2)$. Samples $\{z_j^T\}_{i=j}^M$ drawn from each such distribution thus capture the per-task stochastic factors. To limit the randomness in the learned prior/posterior, we use $M=1$. The latent encoder thus outputs a subtotal of $N*(t+1)$ encodings per input point.

Put together, the global and task-specific latents can be derived as:

$$\begin{aligned}
&\{z_i^G\}_{i=1}^N \sim \mathcal{N}(\psi^G_\mu(s^G),\psi^G_\sigma(s^G)) = \mathcal{N}(\mu_G,\sigma_G^2),\\
&\{z_i^t\}_{i=1}^M \sim \mathcal{N}(\psi^t_\mu(s_i^t,z_i^G),\psi^t_\sigma(s_i^t,z_i^G)) = \mathcal{N}(\mu_t,\sigma_t^2), \forall t \in T,
\end{aligned} \tag{16}$$

where ψ^G and ψ^t are the global and per-task distribution encoders, respectively.

Deterministic Encoder. The deterministic path is similar to that of an ANP [24] where the context projections $\Phi_i^{\text{det}} = \text{MLP}([x_i;y_i])$ form the keys, queries and values for a self-attention operation,

$SA_{\text{det}}(\Phi_i^{\text{det}}, \Phi_i^{\text{det}}, \Phi_i^{\text{det}}) : \Phi_i^{\text{det}} \to r_i$. The resulting order-invariant context representations $\{r_i\}_{i=1}^m$ are fed as values to a subsequent target-to-context cross-attention operation CA_{det}. The keys x_i and queries x_* for CA_{det} come from the context $x_i \in X_{\mathcal{C}}$ and target $x_* \in X_{\mathcal{T}}$ covariates, respectively, *i.e.,* $CA_{\text{det}}(x_i, s_C, x_*) : x_* \to r_*$ where r_* is invariant to the order of context.

Decoder. Different from other NP variants, the NPCL decoder adopts separate decoding mechanisms during training and inference. At train time, we use the available task identity to filter the true N out of $N * (t+1)$ latent path outputs to be processed by the decoder. After this, the decoder concatenates a target input $x_*^t \in X_{\mathcal{T}}^t$ with its N *true* task-specific latents $\{z_i^t\}_{i=1}^N$ obtained from the latent path and its order-invariant feature r_* obtained from the deterministic path thus resulting in N distinct inputs. For $N > 1$ samples of z_t, we first make N copies of x_* and r_* each, and then concatenate these with each z^t. p_θ thus performs the projection $p_\theta([x_*; r_*; \{z_i^t\}_{i=1}^N]) : x \to h_*$ where $x \in \mathbb{R}^{f+2*o}$ and h_* are the logits of an MLP classifier for the target label y_*. We detail the inference-time decoding in section 4.5.

C Experiments and Reproducibility

Configuration. For a fair comparison with the benchmarks of Buzzega et al. [4], we fix the batch sizes for new task's samples and for replay samples to 32 each for the class-IL datasets and to 128 each for the domain-IL datasets. Both the context and target datasets use the same set of augmentations. For S-CIFAR-10, S-CIFAR-100 and S-Tiny-ImageNet, we apply random crops and horizontal flips to both stream and buffer examples following Buzzega et al. [4] and Boschini et al. [3]. For each setting of memory size on each dataset, NPCL adopts the same learning rate (LR) as reported in Buzzega et al. [4] and Boschini et al. [3]. However, NPCL training additionally relies on linearly increasing the learning rate (LR) over a period of 4000 iterations for class-IL and 40 iterations for domain-IL settings. We further apply gradient clipping [35] on L2-norm of NPCL parameters with a cap of 10000.

Hyperparameter tuning. We arrive at the best hyperparameter settings for each of our datasets through grid search over a validation set made of 10% of the training set on each dataset. The search range for number of samples N from the global distribution $\mathcal{N}(\mu_G, \sigma_G^2)$ is $[2, 5, 10, 20, 50, 100]$. Out of these, we found $N = 50$ during training and $N = 10$ during evaluation to perform better in general across all settings.

Similarly, we conducted a grid search over the batch size of the context set $\mathcal{C}$ over the range $[1/16, 1/8, 1/4, 1/2, 1, 1.25]$ of the original (target) batch sizes for each of the dataset. In general, we found that fixing the context batch size to $1/8$ of the target batch size performed better across all datasets. Such context batches are sampled from a context dataset $\mathcal{D}_{\mathcal{C}}^t$ for each task t. $\mathcal{D}_{\mathcal{C}}^t$ is itself created by randomly selecting a subset of the training samples for each class at the beginning of each incremental training task. To decide on the size of the subset for each class, we ran a gridsearch over the range $[50, 100, 150, 200]$ samples per class and found that incorporating 100 random samples per class into $\mathcal{D}_{\mathcal{C}}^t$ performed well across all datasets.

Finally, to decide on the loss weights α, β, γ and δ for D^t, D^G, $\mathcal{L}_{\text{GR}}$, and $\mathcal{L}_{\text{TR}}^t$, we ran gridsearch for each over possible values [0.0, 0.01, 0.05, 0.08, 0.1, 0.15, 0.2, 0.4]. We report the best settings across datasets in Table 7:

	S-CIFAR-10	**S-CIFAR-100**	**S-Tiny-ImageNet**	**P-MNIST**	**R-MNIST**
α	0.05	0.05	0.01	0.1	0.1
β	0.01	0.01	0.01	0.05	0.05
γ	0.2	0.08	0.05	0.1	0.1
δ	0.1	0.1	0.1	0.15	0.15

Table 7: Hyperparameters for loss contributions that were tuned on validation sets for each dataset.

To further ensure reproducibility, we seed the Pytorch-based data loaders using the instructions mentioned at `https://pytorch.org/docs/stable/notes/randomness.html`. All our experiments are then ran using seed values in the range $[0, 9]$.

Method	S-CIFAR-10 *Class-IL*		P-MNIST *Domain-IL*		R-MNIST *Domain-IL*	
oEWC	-91.64		-36.69		-24.59	
SI	-95.78		-27.91		-22.91	
LwF	-96.69		-		-	
$\mathcal{M}_{\text{size}}$	200	500	200	500	200	500
ER	-61.24	-45.35	-22.54	-14.90	-8.24	-7.52
GEM	-82.61	-74.31	-29.38	-18.76	-11.51	-7.19
A-GEM	-95.73	-94.01	-31.69	-28.53	-19.32	-19.36
iCaRL	-28.72	-25.71	-	-	-	-
FDR	-86.40	-85.62	-20.62	-12.80	-13.31	-6.70
GSS	-75.25	-62.88	-47.85	-23.68	-20.19	-17.45
HAL	-69.11	-62.21	-15.24	-11.58	-11.71	-6.78
DER	-40.76	-26.74	-13.79	-8.04	-5.99	-3.41
ANP	-62.80	-49.18	-28.79	-16.44	-12.08	-10.63
ST-NPCL	-46.91	-32.50	-17.03	-12.40	-7.9	-8.11
NPCL (ours)	-39.11	-27.62	-12.81	-8.60	-5.70	-4.10

Table 8: Backward transfer scores for the experiments in Table 1. Best results are in red. Second best results are in blue. All runs of ANP, ST-NPCL and NPCL in the CL settings rely on experience replay (ER).

D Results: Backward Transfer

Table 8 reports the backward transfer for the accuracy scores mentioned in table 1. We further compute the backward transfer based on uncertainty scores to study the effect of forgetting on uncertainty. Fig. 7 shows the correlation between backward transfer of accuracy and uncertainty for the domain-IL datasets P-MNIST and R-MNIST.

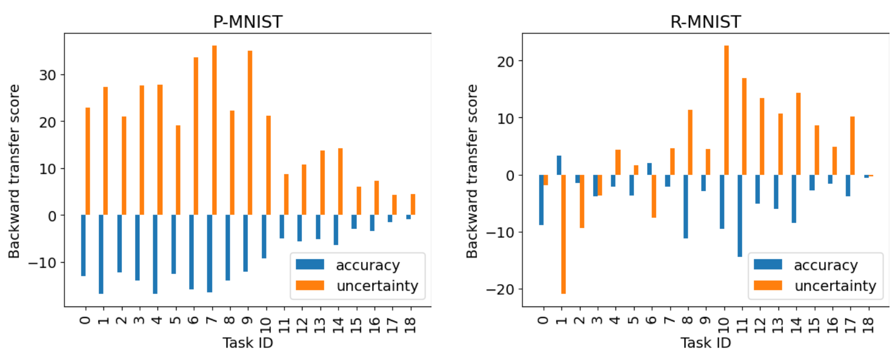
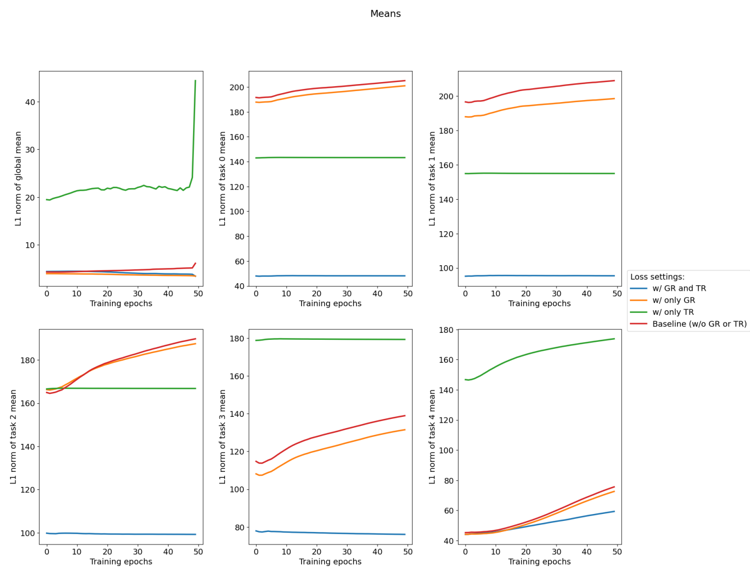


Figure 7: **Backward transfer scores of tasks based on accuracy and uncertainty on domain-IL datasets with** $|\mathcal{M}| = 500$: a higher negative backward transfer on accuracy correlates with a higher positive backward transfer on uncertainty and vice-versa. For better visibility, the uncertainty-based backward transfer scores have been scaled by a factor of 100.

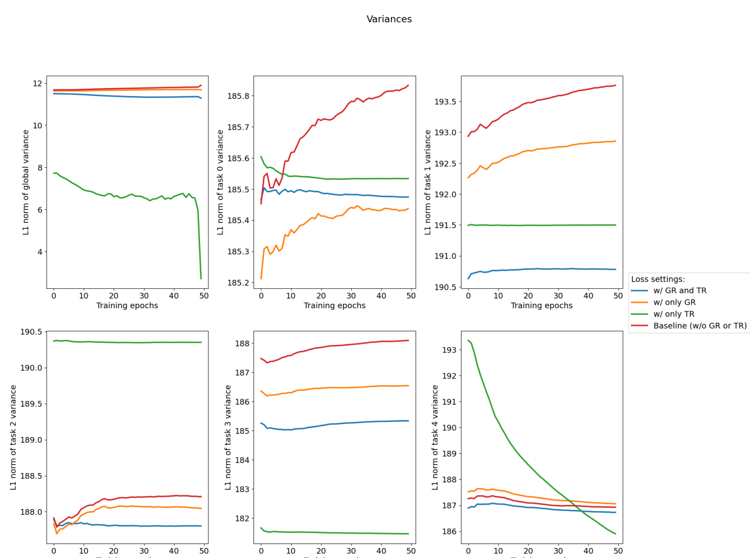
E Ablations

E.1 Effect of regularization on the learned distributions

We record the per epoch L1-norms of global and task-specific means and variances on the last incremental task (task 4) of S-CIFAR-10. As shown in Fig. 8(a) and Fig. 8(b), regularizing the global distribution (GR) alleviates forgetting by limiting the learning of the global and the current task's (task 4) means and variances. This is evident through larger L1-norm of means and smaller L1-norm of variances when GR = 0, *i.e.,* +TR setting. On the other hand, excluding all the objectives, *i.e.,* Baseline NPCL as well as excluding TR from the learning objectives, *i.e.,* +GR setting lead to relatively unstable evolution of the past task means and variances, hence characterizing an increased forgetting. Including both GR and TR in the objective, *i.e.,* NPCL helps find a balance between preserving the global and the past-task distributions while facilitating the learning of the current task distribution.



(a) Effect on the global and task-specific means.

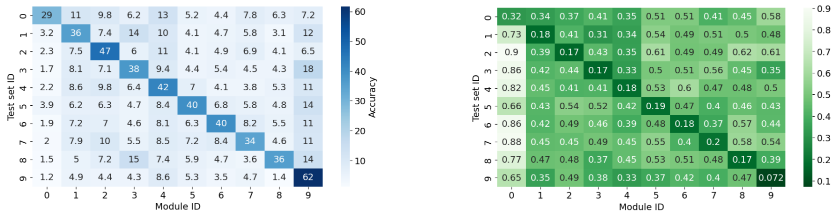


(b) Effect on the global and task-specific variances.

Figure 8: Effect of the proposed global (GR) and task-specific (TR) regularizations on the learning of global and task-specific means and variances during the last incremental training task (task 4) of S-CIFAR-10. NPCL uses both GR and TR while the baseline NPCL uses neither of them.

E.2 How does forgetting effect uncertainty?

Fig. 9 ablates the average accuracies and uncertainties of each task head predictions over the test set of each task at the end of incremental training on S-CIFAR-100. Similar to S-CIFAR-10 (Fig. 4), we observe that the accuracy of predictions made by true task heads are higher than the rest. For predictive uncertainties, the trend is the opposite. Also, more recently trained tasks show lesser forgetting both in terms of accuracy (higher values) and uncertainty (lower values). This generalizes our conclusion on S-CIFAR-10 regarding the outreach of forgetting in CL going beyond accuracy and to other aspects of learning such as the model's predictive confidence.



(a) Accuracy heatmap　　(b) Uncertainty heatmap

Figure 9: Heatmaps depicting the average accuracy and uncertainty of individual task test sets per task head on S-CIFAR-100 with $|\mathcal{M}| = 500$ over an individual run.

E.3 Storage gain of NPCL over DER

Table 9 compares the total episodic memory sizes of NPCL (ours) and DER [4]. We report storage sizes as the dimension of a single 1-d vector constructed by flattening all the vectors that need to be stored by each method in the episodic memory. Namely, NPCL stores $2 * t + 2$ vectors of fixed dimension $\mathbb{R}^{|o|}$ where t is the total number of tasks in a dataset and o is the output size of the mean and variance heads. On the other hand, DER stores $|\mathcal{M}|$ number of logits of dimension $\mathbb{R}^{|N_C|}$ where N_C denotes the total number of classes in a CL dataset. As a result, NPCL has significant storage gains on settings with either large number of classes or a larger memory size. It is worth noting that both NPCL and DER rely on storing original input images and therefore, our comparison does not take the inputs into account.

Method	**S-CIFAR-10**		**S-CIFAR-100**		**S-Tiny-ImageNet**		**P-MNIST**		**R-MNIST**	
$\mathcal{M}_{\text{size}}$	200	500	500	2000	200	500	200	500	200	500
DER [4]	2000	5000	50000	200000	40000	100000	2000	5000	2000	5000
NPCL (ours)	3272	3572	6132	7632	5832	6132	1544	1844	1544	1844
Storage gain (%)	-63.6	**28.56**	**87.746**	**96.184**	**85.42**	**93.868**	**22.8**	**63.12**	**22.8**	**63.12**

Table 9: Storage size comparison of NPCL with DER across different experimental settings of Table 1. Gains are marked in **bold**.

E.4 Out-of-the box novel data identification

Our novel data identification experiments use the S-CIFAR-10 and S-CIFAR-100 datasets interchangeably as $\mathcal{D}_{\text{ID}}$ and $\mathcal{D}_{\text{OOD}}$ given the high degree of similarity between a number of their classes [20].[1] Namely, while evaluating the NPCL trained on S-CIFAR-100, we consider the entire CIFAR-10 test set as $\mathcal{D}_{\text{OOD}}$ whereas the evaluation of the S-CIFAR-10 model treats the test set of first 10 class labels of CIFAR100 to be $\mathcal{D}_{\text{OOD}}$. Further, for an incremental task t, the test sets for $[0, t]$ tasks make up for the ID data $\mathcal{D}_{\text{ID}}$.

As shown in Table 10, the variances computed using either of our proposed metrics on $\mathcal{D}_{\text{ID}}$ are up to a magnitude lower than those on $\mathcal{D}_{\text{OOD}}$. This trend is evident across the incremental evaluation steps even if the differences in the variances between $\mathcal{D}_{\text{ID}}$ and $\mathcal{D}_{\text{OOD}}$ slump with the further arriving tasks. Moreover, for the model trained on the more challenging S-CIFAR-100 setting, we observe that the differences between the $\mathcal{D}_{\text{ID}}$ and $\mathcal{D}_{\text{OOD}}$ variances even grow during the course of incremental training. This implies the potential perks of enabling the inter-task knowledge sharing among NPCL parameters in a CL setup.

[1] The labels for first ten CIFAR-100 classes are the same as `https://huggingface.co/datasets/cifar100` and that for CIFAR-10 classes are the same as `https://huggingface.co/datasets/cifar10`.

Incremental step	CIFAR-100 on S-CIFAR-10 model				CIFAR-10 on S-CIFAR-100 model			
	$\mathcal{D}_{\text{ID}}$ (δ)	$\mathcal{D}_{\text{OOD}}$ (δ)	$\mathcal{D}_{\text{ID}}$ (H)	$\mathcal{D}_{\text{OOD}}$ (H)	$\mathcal{D}_{\text{ID}}$ (δ)	$\mathcal{D}_{\text{OOD}}$ (δ)	$\mathcal{D}_{\text{ID}}$ (H)	$\mathcal{D}_{\text{OOD}}$ (H)
1	$1e^{-6}$	$1e^{-5}$	$9.3e^{-6}$	$8.4e^{-5}$	$1.5e^{-6}$	$8.9e^{-6}$	$1.5e^{-4}$	$1e^{-3}$
2	$2.6e^{-6}$	$1.4e^{-5}$	$6.3e^{-5}$	$2.2e^{-4}$	$1.9e^{-6}$	$5.8e^{-6}$	$5.3e^{-4}$	$1.7e^{-3}$
3	$2.3e^{-6}$	$6.2e^{-6}$	$6.7e^{-5}$	$2.1e^{-4}$	$1.2e^{-6}$	$3.6e^{-6}$	$4.4e^{-4}$	$1.5e^{-3}$
4	$8.1e^{-7}$	$4.8e^{-6}$	$4.6e^{-5}$	$2.2e^{-4}$	$1.1e^{-6}$	$2.5e^{-6}$	$3.5e^{-4}$	$1.2e^{-3}$
5	$7.1e^{-7}$	$1.7e^{-6}$	$4.6e^{-5}$	$1.1e^{-4}$	$8e^{-7}$	$2e^{-6}$	$4.4e^{-4}$	$1.2e^{-3}$
6	-	-	-	-	$6.8e^{-7}$	$1.3e^{-6}$	$4.1e^{-4}$	$8.5e^{-4}$
7	-	-	-	-	$4.e^{-7}$	$1.3e^{-6}$	$3.2e^{-4}$	$8.3e^{-4}$
8	-	-	-	-	$4.9e^{-7}$	$1e^{-6}$	$3.2e^{-4}$	$6.7e^{-4}$
9	-	-	-	-	$3e^{-7}$	$6.9e^{-7}$	$2.5e^{-4}$	$4.7e^{-4}$
10	-	-	-	-	$3.3e^{-7}$	$5.1e^{-7}$	$2.5e^{-4}$	$3.5e^{-4}$

Table 10: Average variances over softmax (δ) and entropy (H) scores of incremental models on in-distribution (ID) and out-of-distribution (OOD) test sets using $N = 50$ samples

E.5 Instance-Level Model Confidence Evaluation

For each target instance x_*, the instance-level model confidence evaluation framework [16] uses the N predictions obtained from stochastic sampling to compute: (a) the prediction interval width (PIW) between the $[2.5, 97.5]$ percentile range of the N predicted classes, (b) the paired two-sample t-test [9] to evaluate the significance of difference between the mean predicted probabilities for the top-2 most predicted classes. As a prerequisite to the latter test, we first verify the normality assumption of the probability differences for NPCL (Fig. 10).

Similar to Fan et al. [9], after computing the PIW per test instance, we split the instances into two groups by the correctness of the majority-vote predictions, obtain the PIW of the true class per instance, and compute the mean PIW of the true class within each group. For t-test evaluation, we compute the mean accuracy per group of the test instances split by their t-test rejection status.

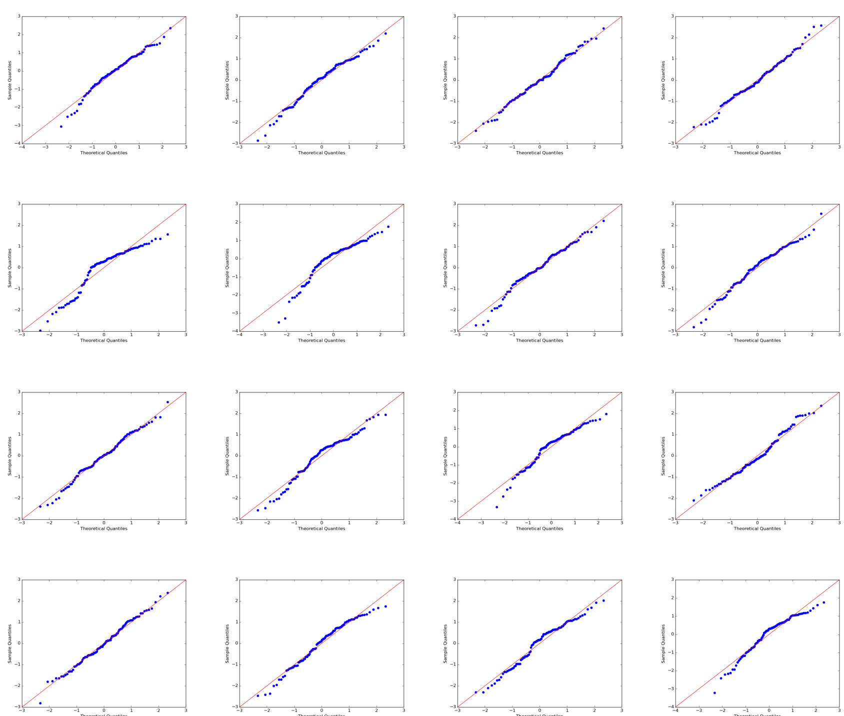


Figure 10: Q-Q plots for the differences in probability between the most and the second most predicted class.

Class	Accuracy	PIW		Accuracy by t-test status	
		Correct	Incorrect	Rejected	Not Rejected
1	82.30	74.17	102.21	83.37	50.00
2	94.00	62.90	79.86	94.07	80.00
3	74.00	54.92	68.48	74.14	64.29
4	71.50	65.42	74.32	72.06	25.00
5	84.80	92.93	106.90	85.37	22.22
6	76.50	75.22	103.58	76.58	60.00
7	94.20	104.9	129.56	94.39	3.00
8	90.50	81.10	127.06	91.12	22.22
9	96.90	72.81	110.86	97.00	66.67
10	96.30	80.60	109.56	96.48	60.00

Table 11: PIW (multiplied by 100) and $t-$test results for classes inferred from their respective task heads after S-CIFAR-10 training.