# CONTINUAL LEARNING WITHOUT CONTINUAL TRAINING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Continual learning requires models to adapt to new domains and new classes while retaining prior knowledge. Many existing methods rely on continued optimization, using regularization, replay, or parameter expansion to prevent new updates from overwriting previously learned knowledge. Instead, we propose replacing continual training with *continual inference*: a PFN-based model that is meta-trained, and then frozen, adapting to new classes only by extending an in-context evidence set. Our model, Latent Concept PFN, performs in-context Bayesian inference over a latent concept space that captures semantic structure shared across domains and classes. As each new domain or class arrives, exemplars are added to the memory; adaptation reflects updated posterior beliefs over latent concepts rather than gradient updates. No parameters are changed, reducing forgetting. The same method handles both domain and class incremental continual learning without task identity. Concept annotations are only used during meta-training, acting as a soft anchor on the latent space rather than a fixed bottleneck. Unlike fixed-vocabulary concept methods, the model also handles noisy, ambiguous, or incomplete annotations by combining concept labels with raw input evidence to discover distinctions beyond the predefined concept set. Experiments on class and domain incremental learning datasets demonstrate competitive continual learning performance while learning interpretable latent concepts.

## 1 INTRODUCTION

Machine learning systems deployed in the real world rarely operate under stationary conditions. Clinical AI models trained at one hospital can fail when deployed elsewhere due to differences in scanners, imaging protocols, and patient populations (Finlayson et al., 2021). In this case the input distribution changes while the underlying prediction task remains the same, giving rise to *domain-incremental* continual learning. In other cases the set of labels itself grows: a diagnostic model may need to recognise a newly defined disease or subtype while still distinguishing the conditions it was originally trained on, giving rise to *class-incremental* continual learning. The two often occur together. During the COVID-19 pandemic, chest X-ray models had to accommodate a new diagnostic class while the scanners, protocols, and patient populations generating the images varied across hospitals and over the course of the pandemic (DeGrave et al., 2021; Roberts et al., 2021). In both settings, models must sequentially adapt to new data while retaining performance on what was seen before, and at test time it is typically unknown which step (and therefore which domain or classes were available) a query comes from. The central challenge is therefore twofold: models must remain plastic enough to learn from new data, while remaining stable enough to avoid catastrophic forgetting, the tendency of gradient-based training to overwrite previously learned representations when parameters are updated on new data (McCloskey & Cohen, 1989; Goodfellow et al., 2015).

Existing continual learning methods approach this challenge through parameter updates. Regularisation methods e.g. Elastic Weight Consolidation (Kirkpatrick et al., 2017), Synaptic Intelligence (Zenke et al., 2017) penalise changes to parameters deemed important for previous tasks, but largely fail when no task identity is given at test time (van de Ven et al., 2022). Replay methods (Rolnick et al., 2019; Buzzega et al., 2020; Chaudhry et al., 2019; Rebuffi et al., 2017) interleave historical data with new examples during training. Class-incremental methods additionally correct the classifier's bias towards recently seen classes (Wu et al., 2019; Zhao et al., 2020) or allocate additional parameters as new classes arrive (Rusu et al., 2022; Mallya et al., 2018; Yan et al., 2021). Methods



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.

built on large pretrained models instead learn a small set of prompts on top of frozen features (Wang et al., 2022b;a) or train a new set of low-ranked parameters when a new task arrives (Liang & Li, 2024; Wu et al., 2025).

More recently, concept-based methods (Yu et al., 2025; Javadi et al., 2026) have been used to provide interpretable structure in continual learning, however these methods still depend on explicit concept supervision and sequential weight updates across new tasks. Although different in mechanism, these approaches all share a common premise: *adaptation requires further optimisation.* As a result, they inherit the fundamental tension between plasticity i.e. learning from new data, and stability i.e. retaining prior knowledge, with catastrophic forgetting arising whenever new updates interfere with previously learned representations. Prototype-based methods are a partial exception: they classify by comparing inputs to class prototypes in the feature space of a frozen pretrained model (McDonnell et al., 2023; Zhou et al., 2025), and so largely avoid updating weights on the stream. However, each class is summarised by a single set of statistics in a fixed feature space, which cannot represent a class whose relationship to the inputs differs across domains.

In this paper, we argue that this framing is unnecessarily restrictive. When domains evolve or new classes arrive, the quantity that must be preserved is *not a specific parameter configuration*, but the conceptual structure that governs the mapping from observations to labels. A rotated digit retains its identity, and a pathology remains characterised by the same morphology despite variation in imaging conditions. Many of these concepts are also shared across classes: the vertical and horizontal strokes that make up handwritten letters also make up digits. From this perspective, continual learning is fundamentally not a problem of preserving model weights, but rather of inferring stable latent concepts from changing observations and relating them to the labels seen so far. This perspective suggests replacing continual training with *continual inference*. If a model can infer the shared latent concepts underlying each new domain or class, then adaptation can occur through belief updating rather than gradient descent. This would avoid parameter overwriting entirely, while allowing knowledge acquired earlier in the stream to transfer naturally whenever latent concepts are reused.

Prior-data fitted networks (PFNs) Müller et al. (2022); Hollmann et al. (2023; 2025) offer a natural starting point. They are trained to approximate the posterior predictive over many datasets drawn from a prior. Although PFNs were designed for prediction on a single, static dataset, we extend this to the continual learning setting and map inputs to a structured latent space anchored to human-interpretable concepts before predicting targets. We meta-train a PFN-based model once, before the continual learning stream, and then keep it frozen. New domains and classes are then learned solely by conditioning on a growing context of stored examples, so no parameters are updated and none can be overwritten.

**Contributions.** **(1)** We introduce Latent Concept PFN, an in-context learner that maps inputs to a structured latent space shaped by concepts and then to targets. **(2)** We show that the Latent Concept PFN achieves competitive continual learning performance with minimal forgetting, without parameter updates on the continual learning stream. **(3)** We show that the structured latent space can be used to explain errors in the model's output. **(4)** We show that the latent space captures information beyond the given concepts and therefore is not restricted to a fixed concept vocabulary.

## 2 Related Work

**Domain-Invariant Representation Learning.** Methods such as Domain-Adversarial Neural Networks (Ganin et al., 2016), Invariant Risk Minimization (Arjovsky et al., 2020), and the Invariant Information Bottleneck (Li et al., 2022) learn representations that are stable across domains, sharing our assumption that domain-invariant latent features exist and are sufficient for prediction. These methods require domain labels and joint optimization across domains to align representations explicitly. In our model, we instead build invariance at meta-training time by anchoring the latent space to concept annotations shared across domains and classes, and by including shift episodes in which inputs are warped while concepts and labels are held fixed, training the encoder to recover the same latent concepts despite changes in appearance. During the continual learning stream no parameters are updated, so invariance is a fixed property of the model rather than something that must be learned at deployment.

**Temporal Domain Generalization.** Temporal domain generalization (TDG) methods characterize distributional drift to extrapolate to unseen future domains (Xie et al., 2023), whereas continual

learning requires retaining performance on all previously seen domains. Existing TDG methods either access multiple source domains simultaneously or fold past domains into gradient updates: DRAIN (Bai et al., 2023), MISTS (Xie et al., 2024), LSSAE (Qin et al., 2022), and EvoS (Xie et al., 2023) all do so. Closest to our setting, Drift-Resilient TabPFN (Helli et al., 2024) also performs no gradient updates at test time, but it is a TDG method evaluated on a single in/out-of-distribution split rather than a sequential stream. It also assumes a fixed label set.

**Concept Bottleneck Models.** CBMs (Koh et al., 2020) predict targets through an intermediate layer of human-interpretable concepts; extensions address incompleteness (Havasi et al., 2022; Shang et al., 2024) and annotation cost (Yuksekgonul et al., 2023; Oikarinen et al., 2023). Recent work applies CBMs to continual learning (Yu et al., 2025; Javadi et al., 2026; Lai et al., 2025), but all still require concept supervision and parameter updates at each step. Our model requires concept supervision only during meta-training and performs no updates on the stream; predictions are made via in-context inference through a latent space that captures the annotated concepts while remaining free to encode discriminative structure beyond them.

**Streaming Continual Learning.** Lourenço et al. (2026) and CURE (Lee et al., 2026) frame continual learning as a context management problem for frozen in-context models, reducing the stability-plasticity trade-off to which examples to keep in memory, a framing our method shares. The key difference is that both operate directly from raw inputs to outputs, whereas our memory lives in a latent concept space shared across domains and classes.

# 3 Continual Learning Without Continual Training

Our goal is a continual learning system that can adapt to new domains and new classes without retraining or replaying past data. The central idea is to use *shared concepts* as a bridge between tasks: the raw appearance of a disease may differ across hospitals, but findings such as pleural thickening remain relevant across both domains and are shared across classes. The Latent Concept PFN learns a latent space anchored to such concepts during meta-training, is then frozen, and adapts during the continual learning stream through in-context inference alone.

**Setup and Model.** Let $\boldsymbol{x}$ denote an input and $y \in \{1, \dots, K\}$ its class label. We introduce a latent vector $\boldsymbol{z} \in \mathbb{R}^d$ and an observed concept vector $\boldsymbol{c} \in \{0,1\}^C$, a noisy, partial annotation of the semantic concepts relevant to $y$ (e.g. *has a curved stroke* for digit images, or *cardiomegaly* for chest radiographs). Predictions for a query $\boldsymbol{x}^{(q)}$ are made given a labeled context set of $n$ examples, $\mathcal{D}_{\text{ctx}} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$, whose inputs $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n)}$ whose inputs form the rows of $\boldsymbol{X}_{\text{ctx}}$ and whose label set we write $\mathcal{Y}_{\text{ctx}} \subseteq \{1, \dots, K\}$. During meta-training the context is part of an episode drawn from the prior (Section 3.3); during continual learning it is the exemplar memory (Section 3.4).

The model makes its prediction in two stages: i) A latent encoder $q_{\boldsymbol{\theta}}$ maps each input to a posterior over a latent $z$ by attending to the other inputs in the context. ii) An in-context classifier $g_{\boldsymbol{\psi}}$, a pretrained TabPFN (Hollmann et al., 2025) that remains frozen, predicts the query label from the latents and the labels in the context. Concept annotations enter only through a separate concept head $h_{\boldsymbol{\phi}}$ (Section 3.2) which shapes $z$ during meta-training and is not used for prediction.

## 3.1 Concept and Label Inference

**Concept Inference.** The latent encoder $q_\theta$ consists of an input embedding network (e.g. a CNN for image inputs) followed by a transformer with the PFN attention mask (Müller et al., 2022) which outputs a Gaussian posterior over the latent $z$ at every position:

$$q_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{X}_{\text{ctx}}) = \mathcal{N}\big(\boldsymbol{z};\ \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{X}_{\text{ctx}}),\ \operatorname{diag}\big(\boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{X}_{\text{ctx}})\big)\big). \tag{1}$$

Under the mask, context positions attend to one another and each query attends to the context only, so queries do not influence each other. The parameters $\boldsymbol{\theta}$ are the only parameters of the predictive path that are trained.

**Label Inference.** The in-context classifier is TabPFN (Hollmann et al., 2025), with parameters $\boldsymbol{\psi}$ that are held fixed throughout. It receives a table with one row per context example, containing the latent $\boldsymbol{z}^{(i)} \sim q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}^{(i)}, \boldsymbol{X}_{\text{ctx}})$ drawn from Eq 1 and its label $y^{(i)}$, together with the query latent

$\boldsymbol{z}^{(q)} \sim q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}^{(q)}, \boldsymbol{X}_{\text{ctx}})$, and returns a distribution over the query label. TabPFN requires labels to be consecutive integers, and assigns one probability to each. We therefore relabel the context with the rank map $\rho_{\text{ctx}} : \mathcal{Y}_{\text{ctx}} \to \{1, \ldots, |\mathcal{Y}_{\text{ctx}}|\}$, which sorts the classes present in the context and replaces each by its position; a context containing classes $\{3, 7, 9\}$ is relabeled $3 \mapsto 1$, $7 \mapsto 2$, $9 \mapsto 3$. Writing $\tilde{g}_{\boldsymbol{\psi}}$ for TabPFN's output distribution over ranks, the distribution over the original classes is obtained by mapping ranks back:

$$g_{\boldsymbol{\psi}}\big(y \mid \boldsymbol{z}^{(q)}, \{(\boldsymbol{z}^{(i)}, y^{(i)})\}_{i=1}^{n}\big) = \begin{cases} \tilde{g}_{\boldsymbol{\psi}}\big(\rho_{\text{ctx}}(y) \mid \boldsymbol{z}^{(q)}, \{(\boldsymbol{z}^{(i)}, \rho_{\text{ctx}}(y^{(i)}))\}_{i=1}^{n}\big) & y \in \mathcal{Y}_{\text{ctx}}, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Classes absent from the context receive no probability mass. During meta-training this order is randomly permuted per episode, Section 3.3, so that the encoder cannot exploit a fixed class ordering.

**Posterior Predictive.** Integrating over the latent $\boldsymbol{z}^{(q)}$ of the query $q$ and of every context example $\boldsymbol{Z}_{\text{ctx}}$ gives

$$q\big(y^{(q)} \mid \boldsymbol{x}^{(q)}, \mathcal{D}_{\text{ctx}}\big) = \int g_{\boldsymbol{\psi}}\big(y^{(q)} \mid \boldsymbol{z}^{(q)}, \{(\boldsymbol{z}^{(i)}, y^{(i)})\}_{i=1}^{n}\big) \times \underbrace{q_{\boldsymbol{\theta}}\big(\boldsymbol{z}^{(q)} \mid \boldsymbol{x}^{(q)}, \boldsymbol{X}_{\text{ctx}}\big)}_{\text{query latent}} \underbrace{\prod_{i=1}^{n} q_{\boldsymbol{\theta}}\big(\boldsymbol{z}^{(i)} \mid \boldsymbol{x}^{(i)}, \boldsymbol{X}_{\text{ctx}}\big)}_{\text{context latents}} d\boldsymbol{z}^{(q)}\, d\boldsymbol{Z}_{\text{ctx}}, \tag{3}$$

where $\boldsymbol{Z}_{\text{ctx}}$ collects the context latents $\boldsymbol{z}^{(i)}$. We estimate Eq. (3) with $S$ joint reparameterised draws of all latents, $\big(\boldsymbol{z}^{(q,s)}, \boldsymbol{Z}_{\text{ctx}}^{(s)}\big)$ for $s = 1, \ldots, S$, averaged in probability space:

$$\hat{q}_S\big(y^{(q)} \mid \boldsymbol{x}^{(q)}, \mathcal{D}_{\text{ctx}}\big) = \frac{1}{S} \sum_{s=1}^{S} g_{\boldsymbol{\psi}}\big(y^{(q)} \mid \boldsymbol{z}^{(q,s)}, \{(\boldsymbol{z}^{(i,s)}, y^{(i)})\}_{i=1}^{n}\big). \tag{4}$$

The same estimator is used during meta-training and at test time, with only the context changing.

**Conditional Independence.** The classifier receives latents and context labels but never the inputs or the concept annotations, so by construction

$$y^{(q)} \perp \big(\boldsymbol{x}^{(q)}, \boldsymbol{X}_{\text{ctx}}\big) \mid \big(\boldsymbol{z}^{(q)}, \{(\boldsymbol{z}^{(i)}, y^{(i)})\}_{i=1}^{n}\big) \tag{5}$$

This is a structural property of the model rather than an assumption about the data. This means that any label-relevant information in the inputs must pass through $z$ and the likelihood term of Eq. (9) pushes whatever the labels require into $z$, including information not captured by the concepts. (Sec. 3.2).

### 3.2 Concept Anchoring during Meta-training

Importantly, the observed concept $\boldsymbol{c}$ enters only through an auxiliary head, and does not play a role in prediction. We attach a separate concept network $h_{\boldsymbol{\phi}} : \mathbb{R}^d \to [0, 1]^C$, a linear map followed by an element-wise sigmoid, to the latents and supervise it to predict $\boldsymbol{c}^{(i)}$ from $\boldsymbol{z}^{(i)}$. We choose the concept vocabulary to be meaningful on both the meta-training data and on the data of the continual learning stream (e.g. vertical stroke or horizontal stroke applies to both letters and digits). The auxiliary loss pulls $\boldsymbol{z}$ towards concept-aligned representations. As intermediate concepts tend to be stable across domains and shared across classes, the concept loss anchors $z$ to structure that does not move when the appearance of x shifts, which we expect to support forward transfer when later domains or classes reuse the same concepts.

Critically, $\boldsymbol{z}$ is not constrained to $\boldsymbol{c}$: the auxiliary loss acts as a soft anchor rather than a hard bottleneck. The concept loss regularises $\boldsymbol{z}$ toward concept space through the BCE term (see Eq. (9)), but does not confine it: the residual dimensions of $\boldsymbol{z}$ remain free to encode discriminative structure beyond the vocabulary. This matters when two classes share a concept code, as several EMNIST letters (App. B). A concept bottleneck model cannot separate such classes, and the concept loss provides no signal to do so. The signal comes instead from the label likelihood, which reaches z through the classifier and rewards any feature of the input that separates the classes. Additionally, the encoder can compare its inputs with the other context inputs through attention. The latent can therefore represent distinctions that the concept vocabulary omits or annotates incorrectly, which we examine directly in Sec. 5.

### 3.3 META-TRAINING FOR CONTINUAL INFERENCE

Meta-training has to prepare a single frozen model for two kinds of change it will meet on the stream: new classes whose meaning must be read from the context (class-incremental), and new domains whose inputs look different but carry the same concepts (domain incremental). We build both into the prior over episodes.

**Source data and episodes.** We meta-train on a source dataset $\mathcal{S} = \{(\boldsymbol{x}^{(j)}, \boldsymbol{c}^{(j)}, y^{(j)})\}_{j=1}^{N}$ with concept annotations. The source data must share the input space and concept vocabulary of the continual learning stream, but need not contain its classes or domains. In our MNIST experiments, the source and stream labels sets are disjoint, $\mathcal{Y}^{\text{src}} \cap \mathcal{Y} = \emptyset$, so everything a model knows about a stream class must reach it through context. Note that due to lack of available data for the CXR datasets, we could not do the same but we instead ensure that no image used in meta-training appears in the stream. Each episode draws a random subset of classes and splits a sample of them into a context set $\mathcal{D}_{\text{ctx}} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{c}^{(i)}, y^{(i)})\}_{i=1}^{n}$ and $m$ queries $\mathcal{D}_{\text{qry}} = \{(\boldsymbol{x}^{(q)}, \boldsymbol{c}^{(q)}, y^{(q)})\}_{q=n+1}^{n+m}$ whose classes all appear in the context. We write $\mathcal{D} = \mathcal{D}_{\text{ctx}} \cup \mathcal{D}_{\text{qry}}$ for the whole episode. In an identity episode, the label is the source class. We add two further episode types.

**Rule episodes (c to y).** In these episodes, the label is a random function of the concept code and to a controlled degree, the pixel inputs. We draw a network $f_\omega$ from the BNN prior of TabPFN (Hollmann et al., 2023), and apply it row by row,

$$y^{(i)} \;=\; \mathrm{bin}\Big(f_\omega\big(\big[\,2c^{(i)} - 1\;;\; \kappa\,(x^{(i)} - \bar{x})\,\big]\big)\Big), \qquad \omega \sim p(\omega), \tag{6}$$

where where $\bar{x}$ is the mean input of the episode and bin sorts the real-valued outputs into $|\mathcal{Y}_{\text{ctx}}|$ label groups by cutting them at $|\mathcal{Y}_{\text{ctx}}| - 1$ of the episode's own outputs. The scale $\kappa$, drawn per episode, sets how much of the rule reads the pixels rather than the concepts. For a proportion of rule episodes, $\kappa = 0$ and the label is a function of $c$ alone. Since the same concept code maps to different labels in different episodes, and the classifier sees only z, a rule episode is answerable only if z carries the concept code and the model reads the mapping from the context. This is what a class-incremental stream demands as new classes arrive.

**Shift episodes (x to c).** In these episodes, the labels and concept annotations are those of the source class and only the inputs change. The episode draws $K \sim \mathcal{U}\{1, \ldots, 5\}$ transformations and assigns each row to one of them. Transformation $k$ is a smooth coordinate map $g_{\xi_k} : \mathbb{R}^2 \to \mathbb{R}^2$ drawn from the same hyperprior as the rules, giving

$$x'^{(i)}(p) \;=\; x^{(i)}\big(p + \tau_k\,(g_{\xi_k}(p) - \bar{g}_{\xi_k})\big), \qquad (x^{(i)}, c^{(i)}, y^{(i)}) \;\mapsto\; (x'^{(i)}, c^{(i)}, y^{(i)}), \tag{7}$$

where $p$ ranges over pixel coordinates. The mean displacement $\bar{g}_{\xi_k}$ is removed so that the content stays in the frame and $\tau_k$ is set so that the mean displacement is a fraction $\alpha_k$ of the mean pixel radius, with $\alpha_k = 0$ for a quarter of the transformations and otherwise log-uniform on $[0.05, 0.5]$. This results in smooth non-linear warps that move strokes rather than scramble pixels so that the transformed input still resemble each other and the targets (Farquhar & Gal, 2018). With probability $0.3$, one transformation is withheld from the context which means that its queries must be answered from a context in which every input was transformed differently. This asks the encoder to recover the same concepts and label from an input whose appearance has changed, which is what a domain-incremental stream demands.

**Prior.** The prior is therefore a mixture of three kinds of episode,

$$p(D) \;=\; \pi_{\text{rule}}\, p_{\text{rule}}(D) \;+\; \pi_{\text{shift}}\, p_{\text{shift}}(D) \;+\; \big(1 - \pi_{\text{rule}} - \pi_{\text{shift}}\big)\, p_{\text{id}}(D), \tag{8}$$

where $p_{\text{id}}$ draws an episode labelled by source-class identity, as described above, and the mixture weights are given in App. E. A rule episode carries unwarped inputs and a shift episode carries class-identity labels, so the two additions never apply to the same episode. With $\boldsymbol{\psi}$ fixed, only the encoder parameters $\boldsymbol{\theta}$ and the concept network parameters $\boldsymbol{\phi}$ are fitted, by minimising

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{\mathcal{D}\sim p(\mathcal{D})}\left[\frac{1}{m}\sum_{q=n+1}^{n+m} -\log \hat{q}_S\big(y^{(q)} \mid \boldsymbol{x}^{(q)}, \mathcal{D}_{\text{ctx}}\big)\right] \\ + \lambda_c\, \mathbb{E}_{\mathcal{D}\sim p(\mathcal{D})}\left[\frac{1}{n+m}\sum_{i=1}^{n+m} \mathbb{E}_{\boldsymbol{z}^{(i)}\sim q_{\boldsymbol{\theta}}(\cdot\mid \boldsymbol{x}^{(i)}, \boldsymbol{X}_{\text{ctx}})} \mathrm{BCE}\big(h_{\boldsymbol{\phi}}(\boldsymbol{z}^{(i)}),\, \boldsymbol{c}^{(i)}\big)\right],\end{aligned} \tag{9}$$

where $\hat{q}_S$ is the estimator (4).

Rule episodes only change the concept-label mapping while the shift episode changes only the inputs. The concept term is applied at every position of the episode, context and query alike, since concept annotations are available for all source examples. After meta-training, $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are frozen. No rules or warps drawn during the continual learning stream.

### 3.4 Continual Inference

During the continual learning stream, no parameter is updated and no concept annotations are used. Continual learning is carried out entirely by conditioning on a growing exemplar memory, which serves as the context. After step $t$, a subset $\tilde{\mathcal{D}}_t \subset \mathcal{D}_t$ of $k$ labeled examples per class is drawn uniformly at random and stored:

$$\mathcal{M}_t = \mathcal{M}_{t-1} \cup \tilde{\mathcal{D}}_t, \qquad \mathcal{M}_0 = \emptyset. \tag{10}$$

The memory stores only input–label pairs and is never otherwise modified. For a test query $\boldsymbol{x}^{(q)}$ after step $t$, the model predicts with (4) using the memory as the context, $\mathcal{D}_{\text{ctx}} = \mathcal{M}_t$ (so $n = |\mathcal{M}_t|$). Predictions are made by $\hat{y}^{(q)} = \arg\max_y \hat{q}_S\big(y \mid \boldsymbol{x}^{(q)}, \mathcal{M}_t\big)$.

In the class-incremental setting, the memory label set is $\mathcal{Y}_1 \cup \cdots \cup \mathcal{Y}_t$, and by Eq. (2) classes that have not yet arrived receive no probability mass. In the domain-incremental setting, the memory mixes exemplars from every domain seen so far. In both cases, the memory is a mixture over steps and hence no task or domain identity is provided. The model must infer the relevant structure from the content of $\mathcal{M}_t$ alone. By default, the whole memory is used as the context. When the memory is larger than the contexts seen during meta-training, we can instead draw $N_{\text{sub}}$ class-balanced random subsets of $\mathcal{M}_t$ of a fixed size, evaluate Eq. (4) on each, and average the resulting predictives in probability space.The separation between a model fixed after meta-training and a memory that grows with the stream is the mechanism by which the model adapts to new domains and classes without retraining

## 4 Experimental Setup

**Datasets.** We run three experiments (summarised in Table 4). Every experiment has two disjoint parts: a source dataset with concept annotations, used for meta-training (Section 3.3), and a continual learning stream, which the model adapts to only through context (Section 3.4). We run two MNIST experiments which share a single source dataset and a single meta-trained model, in which one presents a domain-incremental stream **Rotated MNIST** (LeCun et al., 1998; Lopez-Paz & Ranzato, 2017), stream of five rotations ($0°$, $15°$, $30°$, $45°$, $60°$) over a fixed ten-class label space, and the other a class-incremental stream **Split MNIST** (Zenke et al., 2017; van de Ven et al., 2022), introducing disjoint digit pairs $(0,1), (2,3), (4,5), (6,7), (8,9)$ in sequence. The model is meta-trained on EMNIST letters (Cohen et al., 2017): 26 classes of $28\times28$ grayscale images with $C$=7 binary stroke-attribute concepts with no digit classes seen during meta-training(Cohen et al., 2017). In both, we store $k = 5$ exemplars per class at each step, and after each step we evaluate on every task seen so far. The third experiment is on chest radiographs, **CXR**. We meta-train on a sub-set of VinDr-CXR (Nguyen et al., 2022; 2021; Pollard et al., 2026) with a small slice (1000 images) from TBX11K (Liu et al., 2020). VinDr is labelled with both local findings and global diagnoses. We use $C = 12$ local findings as our concepts (e.g. cardiomegaly, pleural effusion). The stream consists of four hospitals: VinDr-CXR, TBX11K (Liu et al., 2020; 2024), Shenzhen and Montgomery (Jaeger et al., 2014a; Candemir et al., 2014; Jaeger et al., 2014b), and we predict whether or not the scan received a label of tuberculosis. The hospitals differ in various ways such as the scanners available, patient populations and the prevalence of tuberculosis, which is 5.5%, 11%, 50%, and 36% respectively in the queries. At each step we store $k = 16$ exemplars per class, so the memory holds 32 images per hospital and 128 after the last step. Full details are in Appendix C.

**Baselines.** We compare against one replay method, two pretrained-backbone continual learning methods and a standard TabPFN. i) *DER++* (Buzzega et al., 2020) with the replay buffer set to the size of our exemplar memory, ii) *L2P* (Wang et al., 2022b) and iii) *InfLoRA* (Liang & Li, 2024), both on a ViT-B/16 backbone, pretrained on ImageNet-21k and trained for five epochs per task on the full data of each step, and iv) *TabPFN* on frozen ImageNet ResNet-18 (He et al., 2016) features

Table 1: Continual learning metrics on the three datasets, computed from the mean accuracy matrices over 5 seeds. Note that fwt is not defined for class-incremental split mnist and cxr uses balanced accuracy. The top two rows are frozen and condition on the same exemplar memory while the bottom three update parameters on the full data of each step.

| | rotated mnist (domain-il) | | | | split mnist (class-il) | | | cxr tb (domain-il) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| method | aa ↑ | bwt ↑ | fm ↓ | fwt ↑ | aa ↑ | bwt ↑ | fm ↓ | aa ↑ | bwt ↑ | fm ↓ | fwt ↑ |
| Latent Concept PFN (ours) | 0.789 | +0.037 | −0.007 | +0.568 | 0.794 | −0.072 | +0.072 | 0.749 | +0.009 | +0.005 | +0.146 |
| TabPFN | 0.785 | +0.068 | −0.013 | +0.553 | 0.638 | −0.093 | +0.093 | 0.668 | +0.012 | +0.008 | +0.080 |
| DER++ | 0.918 | −0.032 | +0.039 | +0.742 | 0.749 | −0.273 | +0.273 | 0.777 | −0.065 | +0.074 | +0.224 |
| L2P | 0.813 | −0.193 | +0.193 | +0.787 | 0.471 | −0.418 | +0.418 | 0.642 | −0.128 | +0.128 | +0.079 |
| InfLoRA | 0.958 | −0.045 | +0.045 | +0.883 | 0.714 | −0.058 | +0.058 | 0.814 | +0.048 | +0.000 | +0.190 |

projected by PCA to 16 dimensions (size of latent $z$), fitted without labels on our meta-training images and conditioned on the same memory as our method. More details in Appendices F, G.

**Evaluation.** We use standard continual learning metrics computed over the mean accuracy $a_{ij}$ matrices: average accuracy (AA) and backward transfer (BWT) (Lopez-Paz & Ranzato, 2017), the forgetting measure (FM) (Chaudhry et al., 2018) with the final step excluded from the maximum, and forward transfer (FWT) (Lopez-Paz & Ranzato, 2017). To evaluate the concept read-out, we marginalise the concept network $h_\phi$ over the same posterior draws of $z$ that give the class prediction, and score each concept by its AUROC against annotations if (Appendix H).

## 5 Results

**The Latent Concept PFN forgets least in every setting and stays competitive without a single parameter update.** On both domain-incremental benchmarks, forgetting is near-zero (FM $-0.007$ / BWT $+0.037$ on RotMNIST; FM $+0.005$ / BWT $+0.009$ on CXR), with per-task accuracy curves that are flat or slightly rising after introduction (Figs. 3, 3, 1). DER++ and InfLoRA reach higher AA ($0.918$/$0.777$ and $0.958$/$0.814$) by updating parameters on the full data at every step, and InfLoRA alone matches our low forgetting by constraining each step's update to a subspace orthogonal to previous steps' gradients. L2P forgets in every setting, likely because its shared prompt pool is overwritten by later steps. Our method makes no gradient updates, conditioning only on $k$ exemplars per class. Relative to TabPFN, the closest baseline, sharing our backbone, memory, and meta-training data, the AA gap is small on RotMNIST ($+0.004$) but larger on CXR ($+0.081$), suggesting that concept-anchored latents matter most when the shift involves prior shift (TB prevalence varies from 5.5% to 50% across hospitals) rather than pure covariate shift.

**The Latent Concept PFN outperforms baselines in the class-incremental setting.** On Split MNIST, the Latent Concept PFN achieves the highest AA of all methods, outperforming even parameter-updating baselines. L2P and InfLoRA are rehearsal-free, so each step's head is trained on the two new classes alone and inter-task boundaries must emerge implicitly. DER++ sees earlier classes via replay but must re-fit its shared head to a growing label set, risking drift on old boundaries. A context-based model sidesteps this entirely: new classes are incorporated by extending the memory, with no decision boundaries to relearn. Concept anchoring matters most here: the gap over TabPFN is largest on Split MNIST ($+0.156$), suggesting that a compositional latent space allows the model to reuse structure from earlier classes rather than building new representations from scratch, making a handful of exemplars sufficient to distinguish new classes from old ones. The errors that remain tend to be between classes that are close in concept space, which we examine below.

**The concept read-out allows us to explain a drop in accuracy.** We demonstrate this on the largest drop observed across 5 seeds for Split MNIST: accuracy on digits 2 and 3 fell from $0.800$ to $0.560$ (seed 42) when 6 and 7 arrived, with 108 of 320 previously correct images flipping. Crucially, the concept read-out for these images changed by only $0.03$ per concept on average, and no more for the images that became incorrect than those that stayed correct ($0.025$ vs. $0.037$), confirming that the representation itself is stable when new classes arrive. The errors were concentrated: 92 of the 108 flips were 2s predicted as 7s. The lost 2s share a consistent concept profile, where at least 79% read below $0.3$ on curve, loop and symmetric, and above $0.7$ on horizontal, diagonal and two-part: matching the code for the letter 'z'. The stored 7 exemplars occupy the same region: three of the

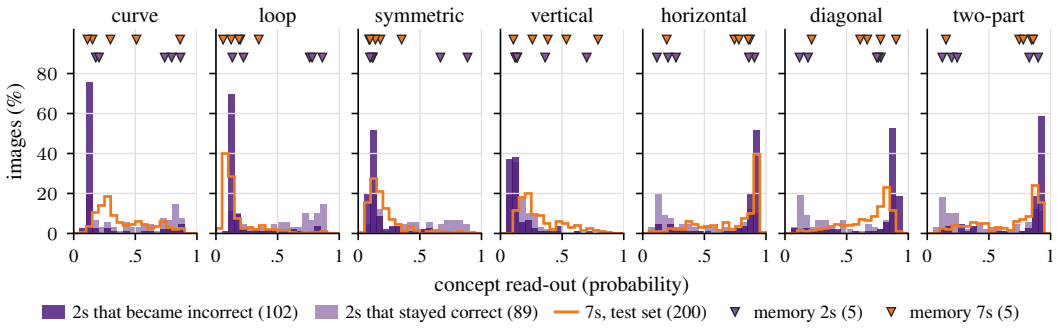



Figure 1: Concept read-out on Split MNIST (seed 42) for 2s when 6 and 7 arrived. We show the 2s that became incorrect, and those that stayed correct alongside the concept-readout for the 7s in the test queries. Triangles mark the memory exemplars of 2 and 7.

five read like a 'z' on all six concepts. Of the 116 correctly classified 2s that also read like a 'z', 72% became incorrect after 7 arrived, against only 25% of the other 2s. The concept read-out therefore identifies not just which images fail, but why: they are 2s whose concept code is indistinguishable from the newly arrived 7 exemplars (Fig. 1).

**The concept set need not be complete as the latent $z$ carries information beyond the concepts.** We measure completeness by replacing $z$ with the concept read-outs $\hat{c}$ as input to stage 2 and adapting the completeness score of Yeh et al. (2020) to measure the ratio of above-chance information in $\hat{c}$ relative to $z$. Scores are $0.97$, $0.93$, and $0.92$ on Split MNIST, CXR, and Rotated MNIST respectively: concepts carry most of the predictive signal, but a gap exists on all three datasets. We focus on RotMNIST, where the gap is largest. Of 25,000 test images, 2,120 are correct under $z$ but not under $\hat{c}$ alone, concentrated on digits 6 ($+0.11$), 3 ($+0.09$), and 5 ($+0.09$). The most common correction is $4 \rightarrow 7$ (108 images): open-topped 4s and nearly-vertical 7s share similar concept codes because the annotation assumes a diagonal stroke for 7 that many MNIST examples lack. To verify that $z$ encodes genuinely different information rather than merely shifting the boundary in concept space, we paired each such 4 with the test 7 at the same angle whose concept read-out was closest (mean difference $0.033$ per concept). $z$ classified both images in a pair correctly in 85% of cases, whereas $\hat{c}$ alone called both a 7 in 88% of pairs. Inspecting individual latent dimensions confirms this: one dimension separates open- from closed-topped 4s, and another separates those open-topped 4s from 7s, a variation in writing style that the predefined concept vocabulary does not capture (Fig. 2).

**Where finding labels exist, we can confirm that the concept read-out is accurate.** On the 30,000 held-out VinDr films, the mean AUROC over twelve findings is $0.76$, reaching $0.89$ for cardiomegaly. For the other hospitals, where finding labels are unavailable, we instead examine cross-hospital consistency. Of the four findings with the largest spread in mean read-out across hospitals, pleural thickening and pulmonary fibrosis are consistently higher in TB-positive patients than TB-negative patients across all four hospitals. The gap is smallest for Shenzhen, which also has the lowest balanced accuracy at the end of the stream (Fig. 5, 3).

## 6 Conclusion and Limitations

We introduced the Latent Concept PFN, which reframes continual learning as continual inference: a model meta-trained with concept annotations to map inputs to a structured latent space $z$ and then to targets $y$, frozen thereafter, and adapted solely by conditioning on a growing exemplar memory. It achieves competitive AA against methods that train on the full data at every step, and outperforms all baselines on class-incremental Split MNIST. Concept anchoring is most beneficial under prior shift (CXR) and in the class-incremental setting. The concept read-out enables post-hoc error analysis, and $z$ encodes discriminative structure beyond the annotated concept vocabulary when needed. The primary limitation is that concept annotations are required at meta-training time, though label-free (Oikarinen et al., 2023) and post-hoc (Yuksekgonul et al., 2023) approaches suggest routes around this via vision-language models for example. Additionally, the concept read-out can help diagnose the source of errors, but it cannot yet intervene on the concepts to correct those errors, as in a conventional CBM.

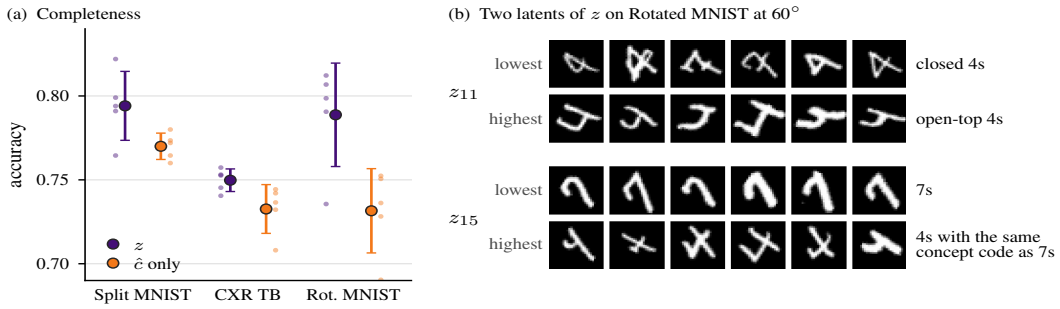



Figure 2: Concept completeness and what $z$ encodes beyond the concepts. (a) Accuracy at the end of the stream when stage 2 is given $z$ or only the concept read-outs $\hat{c}$, for both the memory exemplars and the test images (mean $\pm$ sd over 5 seeds) (b) Rotated MNIST test images at 60° (seed 11) with the lowest and highest values of two latents of $z$. $z_{11}$ separates closed from open-top 4s, and $z_{15}$ separates 7s from 4s that share their concept code.

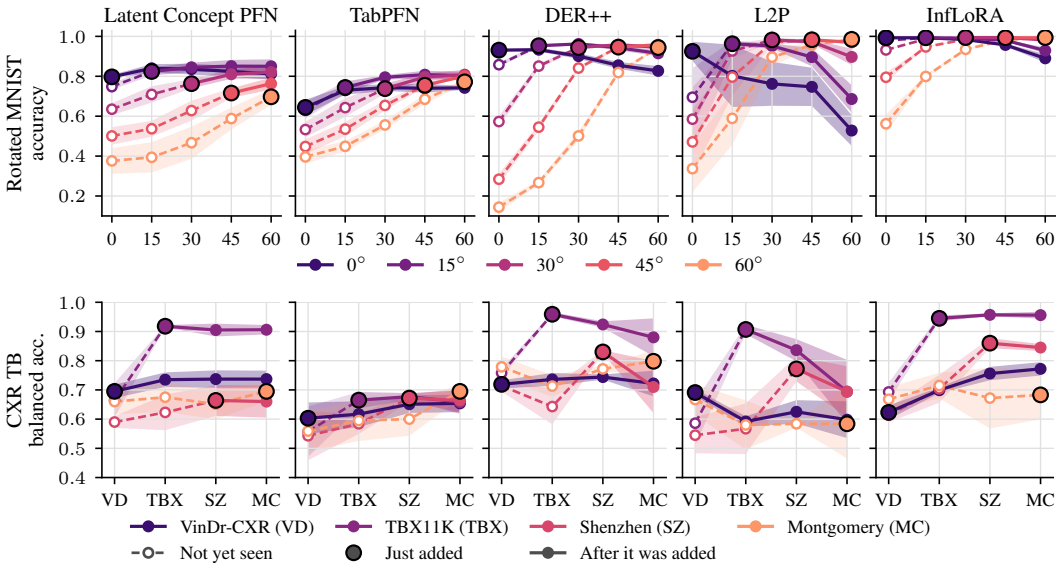



Figure 3: Accuracy on each test domain as domains are added, on two domain-incremental streams: Rotated MNIST (0° $\rightarrow$ 60°, top) and CXR TB (balanced accuracy, bottom), one panel per method. Dashed segments with open markers show forward transfer (domain not yet seen), ringed markers the step at which it is added, and solid segments retention afterwards. Mean $\pm 1$ sd over 5 seeds.

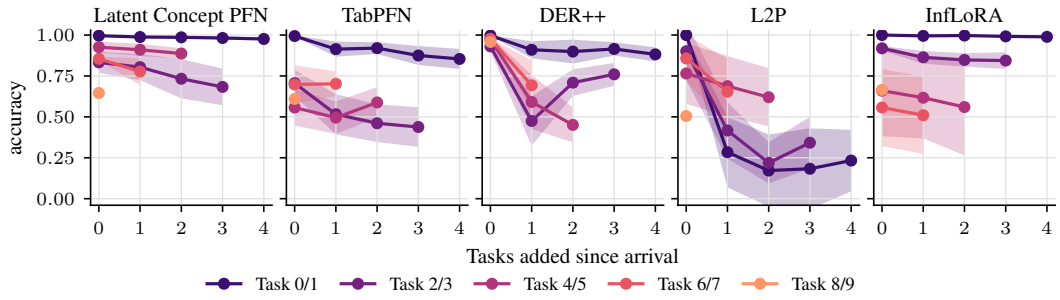



Figure 4: Accuracy on each task of the class-incremental Split MNIST stream ((0,1) $\rightarrow$ (8,9)), one panel per method, aligned at the step the task is added: $x = k$ is accuracy after $k$ further tasks, so a falling line shows forgetting.

## AI USE STATEMENT

In this work, we used generative AI tools for grammar and formatting. We have not used generative AI tools for interpreting results, ideation, generating synthetic datasets, and develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses are not applicable to this work.

Additionally, we used generative AI tools for image placement (better rearranging planned images such as putting the 5 methods in a single row instead of 2x3 setup) to help us comply with the page limit. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

## REFERENCES

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant Risk Minimization, 2020. URL http://arxiv.org/abs/1907.02893. arXiv:1907.02893 [stat.ML].

Guangji Bai, Chen Ling, and Liang Zhao. Temporal domain generalization with drift-aware dynamic neural networks. In *The eleventh international conference on learning representations*, 2023. URL https://openreview.net/forum?id=sWOsRj4nT1n.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *Advances in neural information processing systems*, volume 33, pp. 15920–15930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/b704ea2c39778f07c617f6b7ce480e9e-Paper.pdf.

Sema Candemir, Stefan Jaeger, Kannappan Palaniappan, Jonathan P. Musco, Rahul K. Singh, Zhiyun Xue, Alexandros Karargyris, Sameer Antani, George Thoma, and Clement J. McDonald. Lung segmentation in chest radiographs using anatomical atlases with nonrigid registration. *IEEE transactions on medical imaging*, pp. 577–590, February 2014. ISSN 1558-254X. doi: 10.1109/TMI.2013.2290491.

Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence. pp. 532–547, 2018. URL https://openaccess.thecvf.com/content_ECCV_2018/html/Arslan_Chaudhry__Riemannian_Walk_ECCV_2018_paper.html.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. On Tiny Episodic Memories in Continual Learning, June 2019. URL http://arxiv.org/abs/1902.10486. arXiv:1902.10486 [cs.LG].

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: Extending MNIST to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2921–2926, 2017. doi: 10.1109/IJCNN.2017.7966217. URL https://ieeexplore.ieee.org/document/7966217. ISSN: 2161-4407.

Alex J. DeGrave, Joseph D. Janizek, and Su-In Lee. AI for radiographic COVID-19 detection selects shortcuts over signal. *Nature Machine Intelligence*, 3(7):610–619, July 2021. ISSN 2522-5839. doi: 10.1038/s42256-021-00338-7. URL https://doi.org/10.1038/s42256-021-00338-7.

Sebastian Farquhar and Yarin Gal. Towards Robust Evaluations of Continual Learning, 2018. URL http://arxiv.org/abs/1805.09733. arXiv:1805.09733 [stat.ML].

Samuel G. Finlayson, Adarsh Subbaswamy, Karandeep Singh, John Bowers, Annabel Kupke, Jonathan Zittrain, Isaac S. Kohane, and Suchi Saria. The Clinician and Dataset Shift in Artificial Intelligence. *New England Journal of Medicine*, 385(3):283–286, July 2021. ISSN 0028-4793. doi: 10.1056/NEJMc2104626. URL https://www.nejm.org/doi/full/10.1056/NEJMc2104626.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17:1–35, 2016. URL http://jmlr.org/papers/v17/15-239.html.

Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks, 2015. URL http://arxiv.org/abs/1312.6211. arXiv:1312.6211 [stat.ML].

Marton Havasi, Sonali Parbhoo, and Finale Doshi-Velez. Addressing leakage in concept bottleneck models. In *Advances in neural information processing systems*, volume 35. Curran Associates, Inc., 2022. doi: 10.52202/068431-1699. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/944ecf65a46feb578a43abfd5cddd960-Paper-Conference.pdf.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. pp. 770–778, 2016. URL https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html.

Kai Helli, David Schnurr, Noah Hollmann, Samuel Müller, and Frank Hutter. Drift-resilient TabPFN: In-context learning temporal distribution shifts on tabular data. In *Advances in neural information processing systems*, volume 37, pp. 98742–98781. Curran Associates, Inc., 2024. doi: 10.52202/079017-3134. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b2e2774c8e76afe191b5bf518f5cb727-Paper-Conference.pdf.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: a transformer that solves small tabular classification problems in a second. In *The eleventh international conference on learning representations*, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025. ISSN 1476-4687. doi: 10.1038/s41586-024-08328-6. URL https://doi.org/10.1038/s41586-024-08328-6.

Stefan Jaeger, Sema Candemir, Sameer Antani, Yì-Xiáng J. Wáng, Pu-Xuan Lu, and George Thoma. Two public chest X-ray datasets for computer-aided screening of pulmonary diseases. *Quantitative Imaging in Medicine and Surgery*, pp. 475–477, December 2014a. ISSN 2223-4292. doi: 10.3978/j.issn.2223-4292.2014.11.20. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC4256233/.

Stefan Jaeger, Alexandros Karargyris, Sema Candemir, Les Folio, Jenifer Siegelman, Fiona Callaghan, Zhiyun Xue, Kannappan Palaniappan, Rahul K. Singh, Sameer Antani, George Thoma, Yi-Xiang Wang, Pu-Xuan Lu, and Clement J McDonald. Automatic tuberculosis screening using chest radiographs. *IEEE transactions on medical imaging*, pp. 233–245, February 2014b. ISSN 1558-254X. doi: 10.1109/TMI.2013.2284099.

Amirhosein Javadi, Tuomas Oikarinen, Tara Javidi, and Tsui-Wei Weng. CI-CBM: Class-incremental concept bottleneck model for interpretable continual learning. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=Wf6OpLgj2i.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, March 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/10.1073/pnas.1611835114.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *Proceedings of the 37th international conference*

*on machine learning*, volume 119 of *Proceedings of machine learning research*, pp. 5338–5348. PMLR, 2020. URL https://proceedings.mlr.press/v119/koh20a.html.

Songning Lai, Mingqian Liao, Zhangyi Hu, Jiayu Yang, Wenshuo Chen, Hongru Xiao, Jianheng Tang, Haicheng Liao, and Yutao Yue. Learning New Concepts, Remembering the Old: Continual Learning for Multimodal Concept Bottleneck Models. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 12314–12322. Association for Computing Machinery, 2025. ISBN 979-8-4007-2035-2. doi: 10.1145/3746027.3758157. URL https://dl.acm.org/doi/10.1145/3746027.3758157.

Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, November 1998. ISSN 1558-2256. doi: 10.1109/5.726791. URL https://ieeexplore.ieee.org/document/726791.

Jinmo Lee, Doyun Choi, Moongi Choi, and Jaemin Yoo. Bounded Context Management for Tabular Foundation Models on Stream Learning, 2026. URL http://arxiv.org/abs/2606.18677. arXiv:2606.18677 [cs.LG].

Bo Li, Yifei Shen, Yezhen Wang, Wenzhen Zhu, Colorado Reed, Kurt Keutzer, Dongsheng Li, and Han Zhao. Invariant Information Bottleneck for Domain Generalization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36, 2022. doi: 10.1609/aaai.v36i7.20703. URL https://ojs.aaai.org/index.php/AAAI/article/view/20703.

Yan-Shuo Liang and Wu-Jun Li. InfLoRA: Interference-Free Low-Rank Adaptation for Continual Learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23638–23647, Seattle, WA, USA, June 2024. IEEE. ISBN 979-8-3503-5300-6. doi: 10.1109/CVPR52733.2024.02231. URL https://ieeexplore.ieee.org/document/10658274/.

Yun Liu, Yu-Huan Wu, Yunfeng Ban, Huifang Wang, and Ming-Ming Cheng. Rethinking Computer-Aided Tuberculosis Diagnosis. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2643–2652, June 2020. doi: 10.1109/CVPR42600.2020.00272. URL https://ieeexplore.ieee.org/document/9156613. ISSN: 2575-7075.

Yun Liu, Yu-Huan Wu, Shi-Chen Zhang, Li Liu, Min Wu, and Ming-Ming Cheng. Revisiting computer-aided tuberculosis diagnosis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46:2316–2332, 2024. doi: 10.1109/TPAMI.2023.3330825.

David Lopez-Paz and MarcAurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/f87522788a2be2d171666752f97ddebb-Paper.pdf.

Afonso Lourenço, João Gama, Eric P Xing, and Goreti Marreiros. Bridging Streaming Continual Learning via In-Context Large Tabular Models. *First streaming continual learning bridge at AAAI26*, 2026.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the european conference on computer vision (ECCV)*, September 2018.

Michael McCloskey and Neal J. Cohen. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In Gordon H. Bower (ed.), *Psychology of Learning and Motivation - Advances in REsearch and Theory*, volume 24, pp. 109–165. Academic Press, 1989. doi: 10.1016/S0079-7421(08)60536-8. URL https://www.sciencedirect.com/science/article/pii/S0079742108605368.

Mark D. McDonnell, Dong Gong, Amin Parveneh, Ehsan Abbasnejad, and Anton van den Hengel. RanPAC: Random projections and pre-trained models for continual learning. In *Advances in neural information processing systems*, volume 36. Curran Associates, Inc., 2023. doi: 10.52202/075280-0526. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/2793dc35e14003dd367684d93d236847-Paper-Conference.pdf.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International conference on learning representations*, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

Ha Q. Nguyen, Khanh Lam, Linh T. Le, Hieu H. Pham, Dat Q. Tran, Dung B. Nguyen, Dung D. Le, Chi M. Pham, Hang T. T. Tong, Diep H. Dinh, Cuong D. Do, Luu T. Doan, Cuong N. Nguyen, Binh T. Nguyen, Que V. Nguyen, Au D. Hoang, Hien N. Phan, Anh T. Nguyen, Phuong H. Ho, Dat T. Ngo, Nghia T. Nguyen, Nhan T. Nguyen, Minh Dao, and Van Vu. VinDr-CXR: An open dataset of chest X-rays with radiologist's annotations. *Scientific Data*, 9:429, 2022. ISSN 2052-4463. doi: 10.1038/s41597-022-01498-w. URL https://www.nature.com/articles/s41597-022-01498-w.

Ha Quy Nguyen, Hieu Huy Pham, le tuan linh, Minh Dao, and lam khanh. VinDr-CXR: An open dataset of chest X-rays with radiologist annotations. *PhysioNet*, June 2021. doi: 10.13026/3akn-b287. URL https://doi.org/10.13026/3akn-b287.

Tuomas Oikarinen, Subhro Das, Lam M. Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. In *The eleventh international conference on learning representations*, 2023. URL https://openreview.net/forum?id=FlCg47MNvBA.

Tom Pollard, Benjamin E. Moody, Li-wei H. Lehman, Brian J. Gow, Chrystinne Fernandes, Chen Xie, Alistair Johnson, Roger G. Mark, and Thomas Heldt. PhysioNet as a global platform for biomedical research. *Nature Health*, pp. 792–795, 2026. ISSN 3005-0693. doi: 10.1038/s44360-026-00096-z. URL https://doi.org/10.1038/s44360-026-00096-z.

Tiexin Qin, Shiqi Wang, and Haoliang Li. Generalizing to evolving domains with latent structure-aware sequential autoencoder. In *Proceedings of the 39th international conference on machine learning*, volume 162 of *Proceedings of machine learning research*, pp. 18062–18082. PMLR, 2022. URL https://proceedings.mlr.press/v162/qin22a.html.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, July 2017.

Michael Roberts, Derek Driggs, Matthew Thorpe, Julian Gilbey, Michael Yeung, Stephan Ursprung, Angelica I. Aviles-Rivero, Christian Etmann, Cathal McCague, Lucian Beer, Jonathan R. Weir-McCall, Zhongzhao Teng, Effrossyni Gkrania-Klotsas, Alessandro Ruggiero, Anna Korhonen, Emily Jefferson, Emmanuel Ako, Georg Langs, Ghassem Gozaliasl, Guang Yang, Helmut Prosch, Jacobus Preller, Jan Stanczuk, Jing Tang, Johannes Hofmanninger, Judith Babar, Lorena Escudero Sánchez, Muhunthan Thillai, Paula Martin Gonzalez, Philip Teare, Xiaoxiang Zhu, Mishal Patel, Conor Cafolla, Hojjat Azadbakht, Joseph Jacob, Josh Lowe, Kang Zhang, Kyle Bradley, Marcel Wassin, Markus Holzer, Kangyu Ji, Maria Delgado Ortet, Tao Ai, Nicholas Walton, Pietro Lio, Samuel Stranks, Tolou Shadbahr, Weizhe Lin, Yunfei Zha, Zhangming Niu, James H. F. Rudd, Evis Sala, Carola-Bibiane Schönlieb, and AIX-COVNET. Common pitfalls and recommendations for using machine learning to detect and prognosticate for COVID-19 using chest radiographs and CT scans. *Nature Machine Intelligence*, 3(3): 199–217, March 2021. ISSN 2522-5839. doi: 10.1038/s42256-021-00307-0. URL https://doi.org/10.1038/s42256-021-00307-0.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Greg Wayne. Experience replay for continual learning. In *Advances in neural information processing systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Paper.pdf.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive Neural Networks, October 2022. URL http://arxiv.org/abs/1606.04671. arXiv:1606.04671 [cs.LG].

Chenming Shang, Shiji Zhou, Hengyuan Zhang, Xinzhe Ni, Yujiu Yang, and Yuwang Wang. Incremental residual concept bottleneck models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11030–11040, 2024.

Gido M. van de Ven, Tinne Tuytelaars, and Andreas S. Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4:1185–1197, December 2022. ISSN 2522-5839. doi: 10.1038/s42256-022-00568-3. URL https://www.nature.com/articles/s42256-022-00568-3.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. DualPrompt: Complementary Prompting for Rehearsal-Free Continual Learning. In *Computer Vision – ECCV 2022*, pp. 631–648, Cham, 2022a. Springer Nature Switzerland. ISBN 978-3-031-19809-0. doi: 10.1007/978-3-031-19809-0_36.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 139–149, June 2022b.

Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. SD-LoRA: Scalable Decoupled Low-Rank Adaptation for Class Incremental Learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=5U1rlpX68A.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 374–382, Los Alamitos, CA, USA, June 2019. IEEE Computer Society. doi: 10.1109/CVPR.2019.00046. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00046.

Binghui Xie, Yongqiang Chen, Jiaqi Wang, Kaiwen Zhou, Bo Han, Wei Meng, and James Cheng. Enhancing Evolving Domain Generalization through Dynamic Latent Representations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(14):16040–16048, 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i14.29536. URL https://ojs.aaai.org/index.php/AAAI/article/view/29536.

Mixue Xie, Shuang Li, Longhui Yuan, Chi Harold Liu, and Zehui Dai. Evolving standardization for continual domain generalization over temporal drift. In *Advances in neural information processing systems*, volume 36. Curran Associates, Inc., 2023. doi: 10.52202/075280-0964. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/459a911eb49cd2e0192055ee156d04e5-Paper-Conference.pdf.

Shipeng Yan, Jiangwei Xie, and Xuming He. DER: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 3014–3023, June 2021.

Chih-Kuan Yeh, Been Kim, Sercan O Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. In *Advances in neural information processing systems*, volume 33. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/ecb287ff763c169694f682af52c1f309-Paper.pdf.

Lu Yu, Haoyu Han, Zhe Tao, Hantao Yao, and Changsheng Xu. Language guided concept bottleneck models for interpretable continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 14976–14986, 2025.

Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. In *The eleventh international conference on learning representations*, 2023. URL https://openreview.net/forum?id=nA5AZ8CEyow.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th international conference on machine learning*, volume 70 of *Proceedings of machine learning research*, pp. 3987–3995. PMLR, August 2017. URL https://proceedings.mlr.press/v70/zenke17a.html.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, June 2020.

Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting Class-Incremental Learning with Pre-Trained Models: Generalizability and Adaptivity are All You Need. *International Journal of Computer Vision*, 133:1012–1032, 2025. ISSN 1573-1405. doi: 10.1007/s11263-024-02218-0. URL https://doi.org/10.1007/s11263-024-02218-0.

# A Preliminaries

## A.1 Continual Learning

Continual learning broadly concerns learning from a non-stationary stream of data (van de Ven et al., 2022). Three common scenarios are (van de Ven et al., 2022): (1) *domain-incremental learning*, where the task remains fixed but the data domain changes; (2) *class-incremental learning*, where new classes are introduced over time; and (3) *task-incremental learning*, where a model learns multiple distinct tasks and the task identity is known at test time. Motivated by clinical prediction, we focus on the first two settings. For example, a model deployed across hospitals may encounter different imaging equipment, patient populations, or recording protocols while performing the same diagnostic task. Alternatively, a model initially trained to predict one disease may need to acquire the ability to predict additional diseases. In the latter setting, the disease to be predicted is not known a priori at test time, making task-incremental learning inappropriate.

Formally, a learner observes a stream of $T$ steps. At step $t$ it receives a dataset $\mathcal{D}_t = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n_t} \sim p_t(\boldsymbol{x}, y)$ with label set $\mathcal{Y}_t$, and after step $t$ it is evaluated on all $p_{t'}$ with $t' \leq t$, without access to $t'$.

**Domain-Incremental Learning** The label space $\mathcal{Y}_t = \mathcal{Y} = \{1, \ldots, K\}$ and the underlying task are shared across all steps, but the data distribution $p_t(\boldsymbol{x}, y) = p_t(\boldsymbol{x})\, p_t(y \mid \boldsymbol{x})$ changes from one domain to the next. The change can be in the covariates, $p_t(\boldsymbol{x})$ (covariate shift), in the class prevalence, $p_t(y)$ (prior shift), or in the conditional $p_t(y \mid \boldsymbol{x})$ (concept shift). The goal is to maintain high accuracy on all domains seen so far after observing each new domain. Under concept shift this cannot be achieved by any fixed predictor $f(\boldsymbol{x})$ when the domain is not recoverable from $\boldsymbol{x}$ alone, so predictions must also depend on stored examples.

**Class-Incremental Learning** There are $K$ classes in total, but they arrive sequentially: at first we may only observe two or three classes, and new classes arrive over time. Each step introduces its own classes, so the label sets $\mathcal{Y}_t$ do not overlap and together make up $\{1, \ldots, K\}$. After step $t$ the learner must distinguish between all classes seen so far, $\mathcal{Y}_1 \cup \cdots \cup \mathcal{Y}_t$, without being told which step a test example came from. This is challenging because the decision boundaries change as new classes arrive, and classes from different steps are never observed together.

**Combined Domain and Class-incremental Learning.** It is often the case that we want to deal with both types of shift at once. For example, the COVID-19 pandemic introduced a new diagnostic class for chest imaging while acquisition protocols, patient populations and disease prevalence shifted at the same time (Roberts et al., 2021; DeGrave et al., 2021), and a deployed model is not told which type of change has occurred.

## A.2 Prior-Data Fitted Networks

During meta-training of PFNs (Müller et al., 2022), datasets $\mathcal{D} \sim p(\mathcal{D})$ are sampled from a prior over datasets $p(\mathcal{D})$. For each sampled dataset, one example $(\boldsymbol{x}^{(q)}, y^{(q)})$ is held out as a query and the remaining examples form the context set $\mathcal{D}_{\setminus q}$. The PFN $q_{\boldsymbol{\theta}}$ is trained to approximate the posterior predictive distribution $p(y^{(q)} \mid \boldsymbol{x}^{(q)}, \mathcal{D}_{\setminus q})$ by minimizing the expected negative log-likelihood:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\mathcal{D} \sim p(\mathcal{D})}\, \mathbb{E}_{(\boldsymbol{x}^{(q)}, y^{(q)}) \sim \mathcal{D}} \left[ -\log q_{\boldsymbol{\theta}}\big(y^{(q)} \mid \boldsymbol{x}^{(q)}, \mathcal{D}_{\setminus q}\big) \right]. \tag{11}$$

Minimizing this loss is equivalent to minimizing the expected KL divergence from the posterior predictive distribution to $q_{\boldsymbol{\theta}}$, so a trained PFN approximates Bayesian inference under $p(\mathcal{D})$. At

inference time, labeled examples are provided as context tokens and the model predicts on new queries by conditioning on this context through attention, with no parameter updates. Our method departs from the standard PFN formulation by keeping a pretrained tabular PFN Hollmann et al. (2023; 2025) fixed as the in-context classifier and meta-training only a map from inputs into a latent concept space that this classifier reads, allowing continual adaptation through inference alone.

## B CONCEPTS

Table 2: Concept annotation of the source dataset, EMNIST letters ($C = 7$). A 1 marks a concept that is present in the lowercase form of the letter. The 26 letters have 20 distinct codes; the letters that share a code are $\{a, o\}$, $\{c, u\}$, $\{h, n, r\}$, $\{i, l\}$ and $\{w, x\}$. During meta-training each bit is flipped independently with probability 0.1 whenever it is used.

| | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| curve | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| vertical | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| horizontal | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| diagonal | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| loop | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| two-part | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| symmetric | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |

Table 3: Concept annotation of the MNIST digits, with the same $C = 7$ concepts. It is used only to evaluate the concept read-out and is never given to the model. All ten digits have distinct codes.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| curve | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| vertical | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| horizontal | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| diagonal | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| loop | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| two-part | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| symmetric | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |

## C DETAILED EXPERIMENTAL SETUP

**Source dataset for the MNIST experiments** Both MNIST experiments are meta-trained on EMNIST letters (Cohen et al., 2017), $28 \times 28$ grayscale images in the same format as MNIST. This consists of 26 classes, one per letter of the alphabet, where the letter could be upper or lowercase. There are no digits, so no class of either stream is seen during meta-training. Each letter has an annotation of $C = 7$ binary concepts that cover various stroke attributes such as curve or vertical stroke. The concepts were chosen such that they could apply to digits as well, and a latent space anchored to them during meta-training can carry over to digits. The annotation is a property of the class rather than of the individual image similar to the protocol used in CBMs (Koh et al., 2020). The concepts describe the lowercase form, so it is only approximate for the uppercase images of the same class, and each bit is additionally flipped independently with probability $0.1$ whenever it is used. It is also incomplete, there are 20 distinct codes for the 26 letters. The letters supply images, concept annotations and sometimes a label. The label of an episode is either the identity of the letter or drawn from rule prior in Section 3.3. Note that concept annotations are used only during meta-training, and are not provided during the CL streams.

**Rotated MNIST** The domain-incremental stream is Rotated MNIST (Lopez-Paz & Ranzato, 2017), built on the MNIST handwritten digits (LeCun et al., 1998). Our 5 steps are as follows: rotation by $0°$, $15°$, $30°$, $45°$ and $60°$, with bilinear interpolation applied to the image before normalisation. All ten digits appear in every domain and the label space is shared, so the task is unchanged and only the observation process differs between steps, which is covariate shift. At each step, we draw exemplars from the MNIST training split, rotate by that step's angle, and add it to the memory. For the queries,

we draw from the the MNIST test split instead and rotate. After each step, we evaluate on every domain seen so far with the memory, which consists of a mix of exemplars from all domains seen so far, as the context. No domain id is given to the model and all ten classes are available from the first step onwards.

**Split MNIST** The class-incremental steam is Split MNIST (Zenke et al., 2017; van de Ven et al., 2022), built on the same digits (LeCun et al., 1998). The five steps present the disjoint pairs $(0, 1)$, $(2, 3)$, $(4, 5)$, $(6, 7)$ and $(8, 9)$ in that order, so the label sets do not overlap. The label set grows over our steps. After step $t$ the model must separate all $2t$ classes seen so far and is not told which step a query came from. As in Rotated MNIST, the exemplars are drawn from the MNIST training split and the queries from the MNIST test split, and after each step we evaluate on every task seen so far. The memory grows by ten exemplars per step as we add just the two new digit exemplars for each step to the memory. Predictions are confined to classes seen so far.

**Source datasets for the chest x-ray task** The chest radiograph experiment is meta-trained on VinDr-CXR (Nguyen et al., 2022; 2021; Pollard et al., 2026), in which radiologists labelled every image with both local findings and global diagnoses. We use the local findings as our concepts, selecting $C = 12$ findings with at least 100 positive images in the training split (e.g. cardiomegaly, pleural effusion) and taking the majority vote of the three readers of each training image. As we did for EMNIST, we flip each bit independently with probability 0.1. The diagnoses supply the labels. An episode picks one of five diagnoses (tuberculosis, pneumonia, lung tumour, other disease, no finding) and labels each image by whether it carries that diagnosis, or draws its label from the rule prior of Section 3.3. We also add a slice of 1000 images from the official training split of TBX11L (Liu et al., 2020; 2024), the second hospital in the CL stream. These have a tubercolosis label but no findings, so their concept loss is masked, and they make up 10% of episodes. These are disjoint from every TBX11K image the stream later uses. Unlike the MNIST experiments, the stream's label (tuberculosis) is one of the source diagnoses. All images are greyscale, resized to $224 \times 224$ and z-scored per image. The TabPFN baseline fits its PCA projection on the same 14500 source images, without labels, so it sees exactly the source data our model was meta-trained on.

**Chest x-ray CL stream** This is a domain-incremental stream as the label space does not grow. The stream consists of four hospitals: VinDR-CXR, TBX11K, Shenzhen (Jaeger et al., 2014a; Candemir et al., 2014; Jaeger et al., 2014b), and Montgomery (Jaeger et al., 2014a; Candemir et al., 2014; Jaeger et al., 2014b). We predict whether or not the scan received a label of tuberculosis. The hospitals differ in various ways such as the scanners available, patient populations and the prevalence of tuberculosis, which is 5.5%, 11%, 50%, and 36% respectively in the queries. We therefore use balanced accuracies for all metrics. The queries come from the following: The official VinDR-CXR test split (3000 images, labelled by consensus by 5 radiologists, the official validation split of TBX11K (1794 images), and a random half of Shenzhen (331) and of Montgomery (69). Note that the TBX11K validation split was used instead of test as the test labels are not public. Exemplars come from disjoint pools. At each step we store $k = 16$ exemplars per class, so the memory holds 32 images per hospital and 128 after the last step. No hospital identity is given.

# D DATA

## D.1 MNIST BENCHMARKS

The meta-training prior for the MNIST benchmarks is built from EMNIST-letters, which contains 124,800 handwritten letter images across 26 classes. From this, 1,500 images per letter are randomly sampled (seed 0) and split 80/20 within each class, yielding a **training pool** of 1,200 images per letter (31,200 total) and a **held-out pool** of 300 images per letter (7,800 total). All meta-training gradient steps draw exclusively from the training pool; held-out images are reserved for evaluation episodes only.

**Inputs (x).** The 31,200 training-pool images are 28×28 greyscale, normalised identically to MNIST. Shift episodes warp images on the fly at training time.

**Concepts (c).** Each letter class is assigned 7 binary concept features via a fixed, hand-coded lookup table, so every image of the same letter receives the same concept code. The seven concepts are: *curve, vertical, horizontal, diagonal, loop, two_part, symmetric*. Across the 26 letters, this yields 20

Table 4: The three experiments. The two MNIST experiments use the same source dataset, EMNIST letters (Cohen et al., 2017), and the same meta-trained model. The labels of the source dataset are drawn from the prior rather than taken from letter identity. Classes at each step counts the classes present both in the exemplars stored at that step and in the queries it is evaluated on. $k$ is the number of exemplars stored per class at each step, Eq. 10; the memories end at different sizes because every domain of Rotated MNIST contains all ten digits, while each step of Split MNIST introduces two new ones.

| | **Rotated MNIST** | **Split MNIST** | **Chest radiographs** |
|---|---|---|---|
| Setting | domain-incremental | class-incremental | domain-incremental |
| *Source dataset* | | | |
| Images | EMNIST letters | EMNIST letters | VinDR-CXR (train) + 1000 TBX11K (train) |
| Classes | 26 letters | 26 letters | 5 global diagnoses |
| Concepts $C$ | 7 stroke types | 7 stroke types | 12 radiological findings |
| *Continual learning stream* | | | |
| Images | MNIST digits | MNIST digits | X-ray images from 4 hospitals |
| Steps ($T = 5$) | $0^\circ$, $15^\circ$, $30^\circ$, $45^\circ$, $60^\circ$ | (0,1), (2,3), (4,5), (6,7), (8,9) | VinDr, TBX11K, Shenzhen, Montgomery |
| Classes at each step | 10 | 2 | 2 |
| Exemplars per class $k$ | 5 | 5 | 16 |
| Memory after step $T$ | 250 | 50 | 128 |
| Queries per step | 1000 | 400 | 3000, 1794, 331, 69 |

distinct codes; several letters share a code (a/o, c/u, h/n/r, i/l, w/x). The 10 digit classes each receive a unique code with no collisions. During training, each bit is independently flipped with probability 0.10 before being used as the binary cross-entropy (BCE) target.

**Labels ($\mathbf{y}$).** Depending on the episode type, $\mathbf{y}$ is either the true letter identity or a synthetic label generated by the BNN rule (see Section 3).

**Data volume.** Training runs for 8,000 steps with 32 episodes per step and 64 images per episode, giving 2,048 image draws per step and approximately 16.4 million draws in total, sampled with replacement across episodes.

### D.2 CXR Benchmark

The prior for the CXR benchmark is built from VinDr-CXR (**?**), which contains 18,000 chest radiographs (15,000 official training, 3,000 official test), resized to 224×224 pixels in greyscale. The official training split is itself divided using random seed 0 into $\mathcal{I}_{\text{FIT}}$ (90%, 13,500 films) used for meta-training, and $\mathcal{I}_{\text{HOLD}}$ (10%, 1,500 films) held out entirely. An additional 1,000 films are drawn from the official TBX11K (Liu et al., 2020) training set (stratified by folder, seed 0; 93 positive for TB) and kept strictly disjoint from the TBX11K images used at test time.

**Inputs ($\mathbf{x}$).** 13,500 VinDr films from $\mathcal{I}_{\text{FIT}}$ plus 1,000 TBX11K films, giving 14,500 total.

**Concepts ($\mathbf{c}$).** Twelve radiological findings that each appear in at least 100 positive cases within the VinDr training set are used as concept labels, annotated per image by majority vote of three radiologists. The findings are: *Aortic enlargement, Calcification, Cardiomegaly, Consolidation, ILD, Infiltration, Lung Opacity, Nodule/Mass, Pleural effusion, Pleural thickening, Pulmonary fibrosis,* and *Other lesion*. As with MNIST, each bit is randomly flipped with probability 0.10 for the BCE target. Concept labels are available only for the 13,500 VinDr films; the concept loss is masked on the TBX11K slice.

**Labels (y).** Each episode is framed as a binary classification: one of five diagnoses versus all others. Positive-class counts within $\mathcal{I}_{\text{FIT}}$ are: TB (436), Pneumonia (432), Lung tumor (118), Other disease (3,596), and No finding (9,542). For the TBX11K slice, only the TB label is used.

**Data volume.** Training runs for 2,000 steps with 32 episodes per step and 64 images per episode, giving approximately 4.1 million total image draws.

### D.3 Continual Learning Streams

The continual learning (CL) streams are used solely for evaluation. The model's weights remain **frozen** throughout; only a small set of exemplar images is stored in memory after each task or domain arrives.

#### D.3.1 Split MNIST (Class-Incremental, 5 Tasks)

Digit pairs arrive sequentially: $\{0, 1\}$, $\{2, 3\}$, $\{4, 5\}$, $\{6, 7\}$, $\{8, 9\}$. After each task, 5 exemplars per class are stored from the MNIST training set (10 per task, building to 50 after all five tasks). Query sets contain 200 images per class from the MNIST test set (400 per task, 2,000 in total). The model receives no task identity at inference time and restricts its predictions to classes seen so far via argmax.

#### D.3.2 Rotated MNIST (Domain-Incremental, 5 Domains)

All 10 digit classes appear in every domain, but each domain applies a different rotation to the images: $0°$, $15°$, $30°$, $45°$, $60°$ (bilinear interpolation). Memory stores 5 exemplars per class per domain (50 per domain, building to 250 after all five domains). Query sets contain 100 images per class per domain from the MNIST test set (1,000 per domain, 5,000 in total). The headline evaluation uses the entire accumulated memory as the in-context set. A secondary *matched* protocol averages results over 8 random class-balanced subsets of at most 50 exemplars each.

#### D.3.3 CXR TB (Domain-Incremental, 4 Hospitals)

Hospitals arrive in sequence: VinDr $\rightarrow$ TBX11K $\rightarrow$ Shenzhen $\rightarrow$ Montgomery. The task is always binary: TB versus non-TB. After each hospital, 32 films are stored as exemplars (16 TB, 16 non-TB), building to 128 after all four hospitals; the entire accumulated set serves as the in-context memory. The primary evaluation metric is balanced accuracy; AUROC of the predicted TB probability is reported alongside. The model receives no hospital identity at inference time.

The CXR TB stream passes through four hospitals in sequence. For VinDr, the memory pool is drawn from $\mathcal{I}_{\text{HOLD}}$ (1,500 films, 46 TB-positive), the held-out split that was never seen during meta-trainingand the query set is the official VinDr test set (3,000 films, 164 TB-positive, 5.5% prevalence). For TBX11K, memory is drawn from the official training set minus the meta-training slice (5,480 films, 507 TB-positive), with queries taken from the official validation set (1,794 films, 200 TB-positive, 11.1% prevalence). For Shenzhen and Montgomery, each dataset is split in half by random seed 0: the first half serves as the memory pool and the second half as the query set. Shenzhen contributes 331 films per split (171 and 165 TB-positive respectively, 49.8% prevalence in queries), and Montgomery contributes 69 films per split (33 and 25 TB-positive respectively, 36.2% prevalence in queries).

## E Method Implementation

The model is a three-stage pipeline: (1) a convolutional **image embedder** that maps each raw image to a 128-dimensional vector; (2) a **Stage 1 in-context transformer** that processes a full episode and produces a latent variable $\mathbf{z}$ per image; and (3) a fully frozen pretrained tabular classifier, **TabPFN**, that takes $\mathbf{z}$ as input and produces class predictions. A lightweight **concept head** additionally reads from $\mathbf{z}$ to predict concept labels.

### E.1 Components Shared Across All Datasets

**Stage 1: In-context transformer.** Stage 1 consists of 3 post-LayerNorm transformer blocks with hidden size 256, 4 attention heads, a 4× feed-forward expansion, GELU activations, and zero-initialised output projections. The attention mask follows the PFN convention: context rows attend to each other; query rows attend to context rows; query rows never attend to each other. The embedder's 128-dimensional output is projected to 256 dimensions by a linear layer before entering Stage 1. The combined front-end has 2.41M parameters.

The latent variable $\mathbf{z}$ is 16-dimensional and is drawn from a diagonal Gaussian:

$$q(\mathbf{z} \mid \mathbf{x}, \mathcal{X}_{\text{ctx}}) = \mathcal{N}(\boldsymbol{\mu},\, \mathrm{diag}(\exp(\boldsymbol{\ell}))) \,, \tag{12}$$

where $\boldsymbol{\ell}$ is the predicted log-variance, clamped elementwise to $[-4, 4]$. No KL regularisation term is included in the loss.

**Concept head.** A single linear layer maps from $\mathbf{z}$ (dimension 16) to $C$ concept outputs, followed by a sigmoid. BCE loss is computed on both context and query rows.

**Training objective.** The loss combines a predictive term (negative log-likelihood under TabPFN) with a concept prediction term:

$$\mathcal{L} = -\log\left(\frac{1}{S}\sum_{s=1}^{S} g_{\text{TabPFN}}\left(y_q \mid \mathbf{z}^{(s)}, \mathcal{X}_{\text{ctx}}\right)\right) + \lambda_c \cdot \mathrm{BCE}(\hat{\mathbf{c}}, \mathbf{c}), \tag{13}$$

where $S = 4$ latent samples are drawn jointly during training and $\lambda_c = 1$ in all headline results.

**Episode structure.** Each episode contains 64 rows. The number of context rows $n_{\text{ctx}}$ is drawn uniformly from $\{8, \ldots, 56\}$ and is shared across all episodes in a batch; the remaining rows serve as queries. In 30% of episodes, class counts are balanced; otherwise, exactly one image per class is placed in context and the rest are assigned at random. Each class is guaranteed at least one context row. A random class-to-slot mapping is drawn independently for each episode.

**Stage 2: TabPFN.** We use the raw `PerFeatureTransformer` from `tabpfn==2.0.9`, a pre-trained tabular classifier with 7.24M parameters, 12 transformer layers, and embedding size 192. All weights are completely frozen throughout training and evaluation; only a single forward pass is performed (no ensembling). Class labels are passed in as ranks of the classes present, sorted by class ID. Per-layer activation recomputation is disabled during training.

**Prior mixture.** Each episode is generated as one of three types. With probability 0.40 the episode is a SHIFT episode; otherwise, with probability 0.70, it is a RULE episode; otherwise it is an IDENTITY episode. This gives approximate marginal probabilities of 0.40, 0.42, and 0.18 respectively.

- **Rule episodes.** The synthetic label is generated as

$$y = \mathrm{MulticlassRank}(\mathrm{BNN}([2\mathbf{c} - \mathbf{1};\, s(\mathbf{x} - \bar{\mathbf{x}})])) \,, \tag{14}$$

  where the BNN architecture and hyperprior follow the non-causal branch of TabPFN. The mixing coefficient $\rho$ is set to 0 with probability 0.25 (concept-only rule, no pixels), otherwise drawn from $\log$-Uniform$[0.1, 10]$, controlling the fraction of pixel variance mixed into the input. Up to 10 resamples are performed until every query label appears at least once in the context. The *clean* concept values feed the BNN rule; the *noise-corrupted* concept values are used as the BCE target.
- **Shift episodes.** The number of warp domains per episode is $K \sim \mathrm{Uniform}\{1, \ldots, 5\}$. Each domain applies a smooth spatial warp defined by a BNN coordinate field on a 7×7 grid of control points, bilinearly interpolated to image resolution, with the mean displacement subtracted to prevent global translation. Warp strength $\alpha$ is 0 with probability 0.25; otherwise $\alpha \sim \log$-Uniform$[0.05, 0.5] \times \bar{r}$, where $\bar{r}$ is the mean image radius. With probability 0.30, one domain is withheld from the context, requiring the model to generalise to an unseen warp. Labels and concepts are always those of the original, unwarped image.

**Optimiser and schedule.** We use AdamW with learning rate $3 \times 10^{-4}$ and weight decay $10^{-4}$. The learning rate warms up linearly over the first 10% of steps, then follows a cosine decay to zero.

Gradient norms are clipped to 1.0. Each update step accumulates gradients over 32 episodes via micro-batches.

**Evaluation protocol.** All model weights are frozen. At test time, $S_{\text{eval}} = 128$ latent samples are drawn and their predicted class probabilities are averaged. The entire stored memory (all accumulated exemplars) is used as the in-context set.

**Experimental seeds and metrics.** All experiments are repeated with 5 random seeds (42, 123, 7, 11, 19); results are reported as mean $\pm$ sample standard deviation. Let $a_{ij}$ denote accuracy on task $j$ after training on tasks $1, \ldots, i$. We report:

- **AA** (Average Accuracy): mean of the final row $a_{T,j}$, measuring overall performance at the end of the stream.
- **BWT** (Backward Transfer): $\frac{1}{T-1} \sum_{j<T} (a_{T,j} - a_{j,j})$, measuring how much performance on past tasks has dropped since they were first encountered.
- **FM** (Forgetting Measure, Chaudhry et al. 2018): similar to BWT but relative to the maximum accuracy ever achieved on each task (the final task is excluded from this maximum).
- **FWT** (Forward Transfer): average accuracy on a new domain before any of its exemplars are observed, relative to chance; reported for domain-IL benchmarks only.

### E.2 MNIST-Specific Settings

The same trained model is used for both Split MNIST and Rotated MNIST. The image embedder is a three-layer CNN: three convolutional layers with 32, 64, and 64 channels (3×3 kernels, ReLU activations), followed by two rounds of 2× max-pooling and global average pooling. A final linear layer maps the 64-dimensional pooled representation to 128 dimensions. The embedder has approximately 64k parameters.

Stage 1 uses no dropout. Training runs for 8,000 steps using gradient accumulation over 2 micro-batches of 16 episodes each. Episodes may involve between 2 and 10 classes (the maximum supported by TabPFN). For Rule episodes, the BNN input consists of the 7 concept bits concatenated with all 784 pixel values.

### E.3 CXR-Specific Settings

The image embedder is a ResNet-18 trained from scratch with a single input channel. Each image is normalised per-image using z-score normalisation (no dataset-level statistics). The ResNet output is projected to 128 dimensions and has approximately 11.2M parameters. BFloat16 mixed-precision (`bf16`) is used inside the ResNet to reduce memory usage. No data augmentation is applied.

Stage 1 uses dropout of 0.1. Training runs for 2,000 steps (the final checkpoint is used), with gradient accumulation over 8 micro-batches of 4 episodes each. Training was performed on a single A100 (40 GB), at approximately 2 seconds per step. All episodes are binary (one diagnosis versus all others). Ten percent of episodes are drawn from the TBX11K slice, using only the TB label with the concept loss masked.

For Rule episodes, the BNN input is the 12 radiological findings concatenated with the image average-pooled to $28 \times 28$ (784 values) and z-scored. Rule episodes are always binary; TBX11K-slice episodes are never used as Rule episodes. The spatial warp field uses the same settings as MNIST, applied at $224 \times 224$ resolution.

At inference time, the predicted class is the argmax over {non-TB, TB}. For AUROC computation, the TB probability is renormalised over these two classes only.

## F Baselines Overview

We compare against one replay method, two pretrained-backbone continual learning methods and a standard TabPFN. Aside from the TabPFN and our own method, the other baselines update parameters on the stream.

i) *DER++* (Buzzega et al., 2020) interleaves replayed exemplars with the current step's data and adds a distillation term. Its replay buffer is set to the size of our exemplar memory (50 for Split MNIST, 250 for Rotated MNIST), so the two methods store the same number of examples. However, DER++ performs gradient updates on that buffer while we only use it as context. ii) *L2P* (Wang et al., 2022b) maintains a frozen pre-trained backbone. It learns from a pool of prompts, selecting a subset per input via a key-query match, so adaptations are confined to the prompt parameters. iii) *InfLoRA* (Liang & Li, 2024) adds a low-rank branch per task, constrained to a subspace orthogonal to the gradient space of previous tasks, to a frozen pre-trained backbone. iv) *TabPFN* We pass each image through a frozen ImageNet-pretrained ResNet-18 (cite), get z-scores for its 512-d pooled features and finally project them to 16 dimensions (same size as our latent $z$) using PCA. PCA and z-score are fitted once, without labels, on the same images that we use for meta training. This is then used as input into the same frozen TabPFN that we use for stage 2, using the same memories as our method for context.

Both L2P and InfLoRA use a ViT-B/16 backbone supervised-pretrained on ImageNet-21k and are trained for five epochs per task on the full data of each step. Our method and TabPFN are meta-trained on EMNIST letters and are then frozen, conditioning on only the $k = 5$ exemplars per class held in memory.

# G Baseline Implementation Details

All baselines use the same five random seeds (42, 123, 7, 11, 19), the same query sets, and the same evaluation metrics as our method. No task or domain identity is provided at training or test time.

## G.1 TabPFN on Frozen ImageNet Features

This baseline replaces our learned embedder with a frozen ResNet-18 pretrained on ImageNet (`IMAGENET1K_V1` weights from `torchvision`), operating in evaluation mode throughout. The 512-dimensional global-average-pooled features are extracted without any fine-tuning.

**Input preprocessing.** Images are resized to $224 \times 224$ pixels using bilinear interpolation without centre-cropping, and normalised using the ImageNet channel mean and standard deviation. For MNIST, the MNIST-specific normalisation is first undone before resizing from $28 \times 28$ to $224 \times 224$. For CXR, the 224-pixel greyscale films are divided by 255. In both cases, greyscale images are replicated across three channels.

**Dimensionality reduction.** The 512-dimensional features are z-scored and then reduced to 16 dimensions via PCA, fitted once without labels on the same image pool used for our embedder's meta-training:

- **MNIST**: the EMNIST-letters training pool (31,200 images); PCA-16 explains 52.0% of the variance.
- **CXR**: the VinDr $\mathcal{I}_{\text{FIT}}$ set plus the TBX11K slice (14,500 films); PCA-16 explains 49.5% of the variance.

**Classifier.** We use `TabPFNClassifier` (`tabpfn==2.0.9`) with `random_state` set to the current seed and `n_estimators` = 4 for MNIST and 8 for CXR. The context passed to TabPFN is exactly the stored memory used by our method: 50 exemplars at the end of Split MNIST, the full union of 250 exemplars for Rotated MNIST, and the full union of 128 films for CXR TB. The classifier requires no gradient-based training; it is re-fitted on the current memory at every evaluation step.

## G.2 DER++

DER++ augments standard experience replay with a distillation term that matches current logit outputs on buffer images to the logits stored at insertion time. We use a standard reservoir buffer and apply the method to both MNIST and CXR.

**Loss.** Each training step minimises:

$$\mathcal{L} = \mathcal{L}_{\text{CE}}(\text{stream batch}) + 0.5 \cdot \text{MSE}(f(\mathbf{x}_{\text{buf}}),\, \hat{\mathbf{y}}_{\text{buf}}) + 0.5 \cdot \mathcal{L}_{\text{CE}}(\text{buffer batch}), \tag{15}$$

where $\hat{\mathbf{y}}_{\text{buf}}$ are the logits stored at insertion time. Two independent buffer draws are used per step.

**Optimiser.** AdamW with learning rate $3 \times 10^{-4}$, weight decay $10^{-4}$, gradient norm clipping to 1.0, and batch size 128. Training runs for 15 epochs per task/domain/hospital, with a single optimiser maintained across the entire stream. The buffer capacity is set to match the total memory size used by our method at the end of the stream.

**MNIST (Split and Rotated).** The backbone is our MNIST CNN embedder architecture (3-layer CNN, identical structure, trained from scratch) with an additional `Linear(128, 10)` classification head. No data augmentation is applied. Training data consists of 1,000 MNIST training images per class per task, giving approximately 2,000 images per task for Split MNIST (240 steps per task) and 10,000 rotated images per domain for Rotated MNIST (1,185 steps per domain). The buffer holds 50 exemplars for Split MNIST (10 classes $\times$ 5 exemplars) and 250 for Rotated MNIST (10 classes $\times$ 5 exemplars $\times$ 5 domains). Each step processes 128 stream images plus two independent draws of $\min(128, |\text{buffer}|)$ buffer images. At inference, the argmax is restricted to classes seen so far, matching our method.

**CXR.** The backbone is our CXR ResNet-18 (single input channel, trained from scratch, 128-dimensional output) with an additional `Linear(128, 2)` head. Data augmentation is applied on every training pass: random rotation of $\pm 10°$, scale $\pm 10\%$, and shift $\pm 8\%$; BatchNorm layers are kept in training mode throughout. Training data per hospital is:

- VinDr: 15,000 films ($\mathcal{I}_{\text{FIT}} + \mathcal{I}_{\text{HOLD}}$), $\approx$1,770 steps.
- TBX11K: 6,480 films (5,480 from the training set plus the 1,000-film meta-training slice), $\approx$765 steps.
- Shenzhen: 331 films, $\approx$45 steps.
- Montgomery: 69 films, $\approx$15 steps.

Class-balanced sampling with replacement is used throughout (VinDr is approximately 3% TB positive). The buffer holds 128 films total (32 per hospital $\times$ 4 hospitals); reservoir sampling determines which films are retained as the stream progresses.

### G.3 L2P AND INFLORA (CXR ONLY)

Both Learning to Prompt and InfLoRA (Liang & Li, 2024) use a ViT-B/16 backbone pretrained on ImageNet-21k (`vit_base_patch16_224.augreg_in21k` from `timm`), which remains completely frozen throughout. Greyscale CXR images are resized to $224 \times 224$, replicated across three channels, and scaled to $[0, 1]$; ImageNet normalisation is not applied. Training data per hospital matches the DER++ CXR setup above. No rehearsal buffer is used.

**L2P.** A prompt pool of $M = 10$ prompts of length $L_p = 5$ tokens is maintained. At each forward pass, the top $N = 5$ prompts are selected by key-query matching, with key-matching loss weight $\lambda = 0.5$. The trainable parameters are the prompts, the matching keys, and a `Linear(768, 2)` classification head. Training uses Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$) at learning rate $3 \times 10^{-3}$ ($1 \times 10^{-3}$ for split mnist), batch size 128, fp16 AMP, and cross-entropy loss, for 5 epochs per hospital with a single optimiser maintained across the stream.

**InfLoRA.** An InfLoRA branch of rank $r = 10$ is applied to the `qkv` projection of every transformer block (12 in total). After each hospital, the matrix $B_t$ is set to the top-$r$ left singular vectors of the layer inputs after projecting out the accumulated DualGPM subspace; only the matrix $A_t$ is trained. At the end of each hospital, $A_t B_t$ is merged into the frozen weights and the DualGPM subspace is updated. The DualGPM threshold increases linearly: $\varepsilon_t = 0.95 + 0.05 \cdot t/T$ for hospital $t \in \{1, \ldots, T\}$. The subspace is computed from 32 randomly sampled training images per hospital (seeded). The trainable parameters are $A_t$ and a `Linear(768, 2)` head; a fresh Adam optimiser ($\beta_1 = 0.9$, $\beta_2 = 0.999$) is used per hospital at learning rate $5 \times 10^{-4}$, batch size 128, fp16 AMP, and cross-entropy loss, for 5 epochs per hospital.

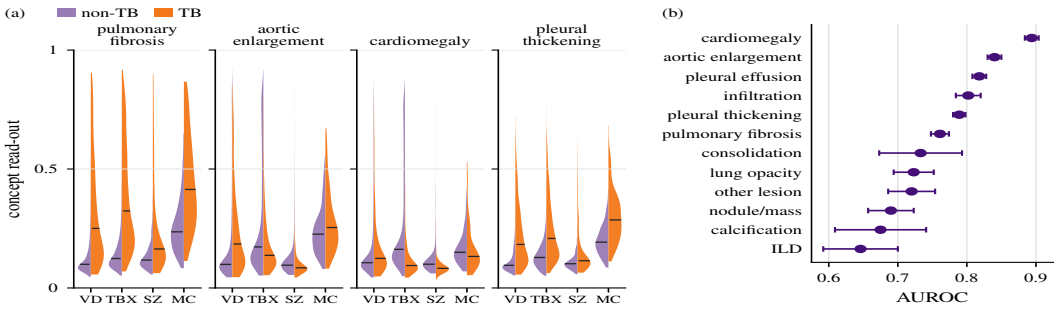



Figure 5: Concept read-out on the chest x-ray stream. (a) distribution of the read-out $\hat{c}$ over the query films of each hospital, split by tuberculosis label, for the four findings whose mean read-out differs most across hospitals. (b) read-out AUROC per finding on the vindr-cxr test split mean $\pm$ sd over five seeds.

## H Evaluation

**Interpreting the latent space** The concept network $h_\phi$ (Section 4.2) gives an independent probability for each of the concepts. Since $h_\phi$ is trained on posterior draws of $z$ (Eq. (8)), we marginalise the latent in the same way as the label predictive (Eq. (5)). For a query $x^{(q)}$ with the memory $\mathcal{M}_t$ as context,

$$\hat{c}^{(q)} \;=\; \mathbb{E}_{z\sim q_\theta(\cdot|x^{(q)},X_{\mathrm{ctx}})}\big[h_\phi(z)\big] \;\approx\; \frac{1}{S}\sum_{s=1}^{S} h_\phi\big(z^{(q,s)}\big),$$

reusing the $S$ draws that give the class prediction, so that $\hat{c}_k^{(q)}$ is the predicted probability that concept $k$ is present. To evaluate the read-out, we annotate the ten digits with the same seven concepts (Table 3). These annotations are used only for scoring and never reach the model. We score each concept by its AUROC against the annotation. When describing where an input lies in latent space, for example how far its latent moves when a step arrives, or which memory exemplar or held-out letter image (never used in meta-training, with its own letter episode as context) is nearest to it, we use the posterior mean $\mu_\theta(x^{(q)}, X_{\mathrm{ctx}})$. For CXR, we only have concept annotations for the VinDr-CXR set.

**Metrics** We use standard continual learning metrics computed over the mean accuracy $aij$ matrices. i) Average accuracy (AA) (Lopez-Paz & Ranzato, 2017), computed as the mean of the final row in the $aij$ matrices. $\mathrm{AA} \in [0, 1]$, higher is better. ii) Backward transfer (BWT) (Lopez-Paz & Ranzato, 2017), where $\mathrm{BWT} \in [-1, 1]$, higher is better. iii) Forgetting Measure (FM) (Chaudhry et al., 2018) with the final step excluded from the maximum, where $\mathrm{FM} \in [-1, 1]$, lower is better. iv) Forward Transfer (FWT) (Lopez-Paz & Ranzato, 2017), $\mathrm{FWT} \in [-\frac{1}{C}, 1 - \frac{1}{C}]$, higher is better. We do not measure this for Split MNIST as in the class incremental case, the model has not been trained to output $j$'s classes before task $j$ arrives. FWT usually subtracts the accuracy of a randomly initialised model. As our method and most of our baselines build on pretrained components, we instead subtract chance ($1/classes$ or $0.5$ for balanced accuracy in cxr).

## I Additional Figures

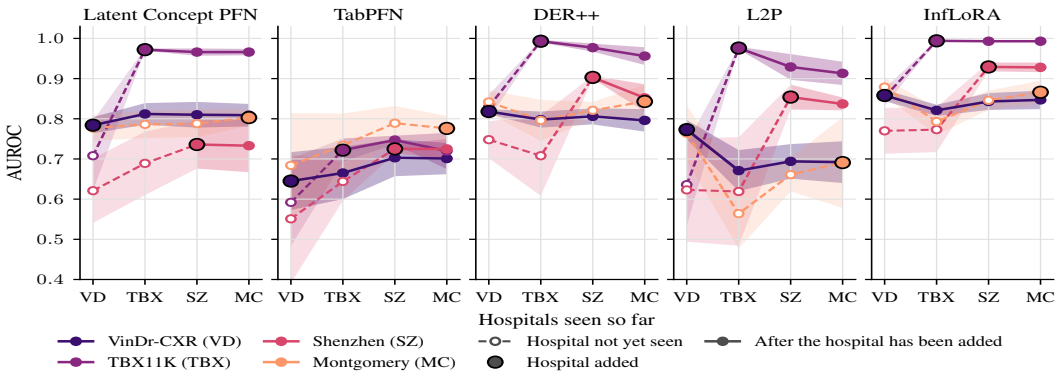



Figure 6: AUROC on each test hospital along the cxr tb stream, one panel per method (mean ± sd over 5 seeds). balanced-accuracy version in figure 3.

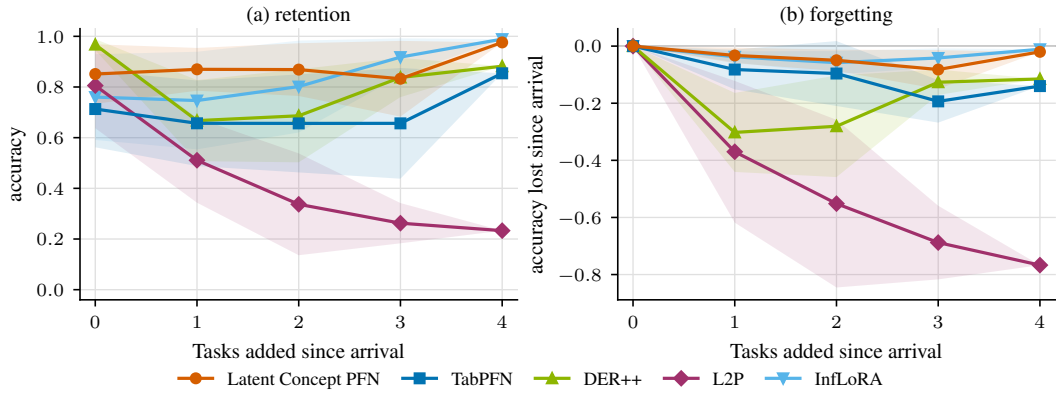



Figure 7: Retention on the class-incremental Split MNIST stream, as a function of the number of tasks added since a task arrived. (a) Accuracy on the task; (b) accuracy lost relative to the step at which the task arrived, so each task's own difficulty cancels out. Over 5 seeds; shaded bands are $\pm 1$ sd across those tasks (a single task at $k = 4$).