Revisiting continual learning through the unsupervized learning of dynamic contexts

Anonymous authors
Paper under double-blind review

Abstract

Continual learning from non-stationary data streams remains a fundamental challenge for autonomous agents. We revisit continual learning by proposing a formal framework that addresses the dual challenge of context alternation and within-context drift. We illustrate this on context-aware anomaly detection, where detecting anomalies alone is insufficient: the system must determine whether detected anomalies are hazardous within specific, evolving contexts. To address this, we introduce a benchmark with alternating contexts and smooth distribution drifts, and propose a solution leveraging topology-preserving vector quantization (GNG-T) to simultaneously track context-specific representations and register so-called contexts. Contexts encode both a structural characterization of the current distribution and context-dependent classifiers. Experimental results demonstrate that explicit context recognition significantly improves recovery after context switches and enables effective anomaly hazard classification, outperforming approaches that continuously train a single classifier. The code for the experiments is available at [**URL OMITTED FOR BLIND REVIEW**].

1 Introduction

In numerous real-world applications, particularly those involving autonomous agents or embedded perception systems, data encountered by the system over time cannot be assumed to follow the classical i.i.d. (independent and identically distributed) hypothesis that underlies most machine learning models. Instead, data are often sequential, task-dependent, and sometimes subject to temporal drift. This situation is a fundamental challenge for adaptive systems: how to continuously extract information and learn from input streams that are subject to sudden changes and not globally stationary.

This challenge is closely related to the "General Continual Learning" (GCL) in the survey Wang et al. (2024) and lifelong machine learning in Parisi et al. (2019). As highlighted in these reviews and more specifically in Wang et al. (2024), most of the litterature on continual learning is oriented toward catastrophic forgetting in deep learning architectures. There, the main challenge is to tackle a supervized learning problem when the data distribution is revealed incrementally.

However, in their survey, Parisi et al. (2019) point out that continual learning is not only about learning static tasks that are discovered incrementally, but also and mainly about an autonomous agent learning from a continuous stream of data that changes with time. By construction, an autonomous agent interacts online with its environment and has to learn, continuously, from the stream of data it receives while behaving. A behavior is, by definition, contextualized : a specific action may be relevant in one context and not in another.

In this paper, we thus propose to go back to the original inspiration of continual learning : an autonomous agent learning from a continuous stream of contextualized data which appear in a statistical context that can drift through time. While several paradigms have been proposed to address similar situations (including lifelong learning, task-incremental learning, and continual learning) few provide the structural flexibility required to jointly address the dual challenge of context alternation and within-context drift. In the literature on continual learning, these contexts are either called tasks or contexts, we will stick to the word contexts

in the rest of the paper. Note that the context is not explicit in the data stream, it refers to a distribution of inputs that changes from one context to another.

Our first contribution is to reconsider continual learning by introducing a general formal framework encompassing both previously proposed frameworks (van de Ven et al., 2022; Wang et al., 2024; Parisi et al., 2019) as well as offering new perspectives, especially in a learning situation based on a continuous stream of data. To illustrate the various aspects covered by our formalised framework, we then choose to focus on a specific type of problem that exacerbates the difficulties raised by online learning of non i.i.d. data observed in alternating contexts which distributions are not stationary. In such situations, the problem becomes even more critical when the goal is not simply to perform classification or regression, but to detect and react to anomalies. Such rare or unexpected observations diverge significantly from the learned regularities, thus their detection lies in their statistical categorization as outliers. However, not all anomalies are equally meaningful. In safety-critical domains, such as autonomous driving, industrial inspection, or medical monitoring, the system must be able to identify anomalies that should trigger significant warning signals: observations that are not only statistically unexpected, but that are also hazardous within specific contextual configurations. This question of interpreting the data in the context in which these are produced is not present in the formalism of Wang et al. (2024). For such context-aware anomaly detection problems, the notion of alarming situation is intrinsically tied to the context-dependent and temporally-evolving structure of the data distribution. An autonomous agent facing this problem will handle a continuous stream of data that are ambiguous with respect to the final decision. Making sense of these data requires to consider them in the current context and to identify this context which is not explicitely provided. Moreover, we posit that context-aware anomaly detection problems involve two sources of non-stationarity. First, the data stream alternates between different contexts, each governed by its own underlying distribution. Second, even when restricted to a single context, the generative distribution is not static but gradually evolves due to environmental drift, sensor aging, wear of materials, or behavioral adaptation. The learning system must therefore be able to distinguish between sudden shifts of context and smooth changes of the current context. The different contexts must be somehow memorized (thus avoiding catastrophic forgetting) while each context must be known through its current distribution (thus avoiding to rely on an outdated long-term history, that would increase the ambiguity of data).

Our second contribution is to propose an artificial benchmark for evaluating models that simultaneously address the various challenges posed by non i.i.d. and non stationary stream of data. Following the above discussion about context-aware anomaly detection, our benchmark consists of 2D data drawn from various context-specific distributions, with two types of outliers : benign and alarming. Let us nevertheless recall that context-aware anomaly detection is not the topic of our paper: we choose it as an application framework solely to illustrate the different aspects of our formalism and in order to instantiate the questions which arise in this framework.

Our third contribution in this paper is a model that is able to track and store representations of context-specific distributions in a fully unsupervised way. This model is based on a combination of neural networks that perform topographical vector quantization. We show how this model fulfills the different requirements that make it able to deal with our benchmark data for context-aware anomaly detection. One claim of this paper is that such topology-preserving vector quantization methods are well suited for this problem.

After detailing the works related to our contributions in Section 2, we present our generalized continual learning formalism in Section 3. Section 4 presents the benchmark we propose to validate a model's ability to address a general continual learning problem. Section 5 shows that the various types of classical continual learning are indeed instances of our formalism, using the nomenclature of Section 3 as a basis. We propose a model for anomaly detection in Section 6, before detailing the experimental results in Section 7 and concluding.

2 Related Work

Among the famous continual learning methods, Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) constrains the weights crucial to maintain performance on previous tasks when processing inputs from new tasks. Synaptic Intelligence (SI) (Zenke et al., 2017), although similar in principle to EWC, computes the

importance of weights with regards to previous contexts online. These two methods assume the availability of context identities during training and testing. Parallel to these two methods, context dependent gating (XdG) (Masse et al., 2018) gates a random subset of the model parameters based on the context in which the input was presented. Used in tandem with parameter regularization methods (e.g. EWC and SI), it can alleviate catastrophic forgetting on an increasing number of tasks. However, much like synaptic stabilization methods, XdG requires the availability of context identities beforehand. In order to remove this constraint of prior knowledge of the contexts, Masse et al. (2018) stated that it is necessary to have algorithms that can identify the current task or context, compare it to previously learned contexts and then perform the gating based on this comparison. The first two requirements belong to the general framework we are trying to develop in this paper. In this same framework of task-incremental learning, functional regularization methods restrict the models' output in order not to change too much on a set of anchor points. For example, Learning without Forgetting (LwF) (Li & Hoiem, 2017) constrains the output of the model executed on input from the current task not to change for parameters crucial for performance on old tasks. On the other hand, Functional Regularization Of Memorable Past (FROMP) (Pan et al., 2020) regularizes the network outputs at a few memorable past examples that are crucial to avoid forgetting. Both methods are applied on incremental image classification problems and none of them handles online input streams, unlike Chaudhry et al. (2019) in which the algorithm processes the input data stream during a single pass. They developed an algorithm (A-GEM) based on Lopez-Paz & Ranzato (2017) which performs comparatively well to the parameter regularization methods cited above. Nevertheless, despite the fact that their continual learning definition is more general as it happens in a one pass of the data stream, it still assumes we are provided with the identities of task information both during training and testing instead of learning it from the data itself as we suggest in our work. One missing aspect in most continual learning algorithms is the ability to deal with non-stationary distributions that might undergo a concept drift (Widmer & Kubat, 1996). The assumptions of these methods is that the knowledge learned from a specific context remains permanently valid. This hypothesis does not hold in dynamic environments where contexts can evolve through time. Gaussian Residual Scoring (GRS) (Kurle et al., 2019) is one of the few methods that take concept drift into consideration during training. GRS adapts to non-stationary distributions with underlying drift via imposing Bayesian exponential forgetting or through a diffusion process applied to the neural parameters. This approach to continual learning is closer to the definition we put forth in this paper. However, their methods are not exactly online. Depending on the dataset that was used in the experiment, the samples their model can see at an instant t range between 10 and 1000 and it needs to be fed 10 times more samples at the beginning when $t = 0$ on almost all datasets. Reactive Subspace Buffer (RSB) (Korycki & Krawczyk, 2021) is another method that bridges the gap between the fields of continual learning and concept drift adaptation. RSB is a centroid driven memory buffer capable of tracking drift in previously learned classes by monitoring adaptive clusters, allowing label switching and splitting. To make the set-up more challenging, the method was evaluated on a binary recommendation problem where sub-classes (e.g. the identity of an input image in a classification setting) were omitted and only the super-classes (0/1) that determine whether the user likes or dislikes an input were retained. It is worth noting that the type of drift we tackle in this paper affects input distributions themselves instead of their labels, which is different from the case Korycki & Krawczyk (2021) addressed. Another method that adapts to concept drift in non-stationary environments is Learn++.NSE (Elwell & Polikar, 2011). It is an ensemble-based method that incrementally learns from sequential batches of data. This algorithm is able to learn new knowledge, temporarily forget irrelevant knowledge, and then recall it when it is relevant again.

In this paper, we suggest tackling context agnostic continual learning via self-organized networks due to their suitability for online training and their modest resource consumption. Several prior works leveraged these methods to alleviate catastrophic forgetting. SOMLP (Bashivan et al., 2019) uses a SOM (Self-Organizing Map) to gate inputs towards specific nodes of a Multi-Layer Perceptron (MLP). It was tested in an incremental image classification setting and it displayed competitive performance compared to existing continual learning paradigms like EWC (Kirkpatrick et al., 2017) and GEM (Lopez-Paz & Ranzato, 2017) when memory resources were limited. However, SOMLP requires a pretraining of the SOM layer on a portion of the training dataset. Parisi et al. (2018) realized continual object recognition using the growing dual-memory (GDM), which is an architecture comprised of a deep convolutional feature extractor and two self-organizing networks based on the Gamma-GWR model (Parisi et al., 2017). Even though GDM outclasses the previous

best scores on the chosen benchmark dataset, it requires a convolutional feature extractor that is pretrained offline. Moreover, one of the self-organizing networks uses class labels as task-relevant signals during training. Gepperth & Karaoguz (2016; 2017) used a SOM in conjunction with linear regression to tackle an incremental image classification problem in which their model is trained on all classes except one. Afterwards, the data corresponding to the unseen class is presented to the algorithm. At the beginning of training, there is an initialization phase in which the model makes multiples passes over the initial input data, before being presented with the excluded class. Moreover, the instant at which the system starts processing the left out class is known and this information was used to adapt the model parameters. Masuyama et al. (2019; 2022; 2026; 2023) used a correntropy-induced metric (CIM) (Chalasani & Principe, 2015) in tandem with self-organization to handle continual learning scenarios. They developed multiple CIM-based self-organizing models, e.g. TCA (Masuyama et al., 2019), CAEA (Masuyama et al., 2022), CAE (Masuyama et al., 2026) and CAEAC (Masuyama et al., 2023), that perform well on class-incremental –both supervised and unsupervised– learning scenarios compared to SOINN-based algorithms, e.g. ASOINN (Shen & Hasegawa, 2008), SOINN+ (Wiwatcharakoses & Berrar, 2020) and GSOINN+ (Wiwatcharakoses & Berrar, 2021). Both families of algorithms –CIM- and SOINN-based– can handle class-incremental learning scenarios well, but they're not designed to deal with concept drift as we define it in this paper. Moreover, CIM-based algorithms require the computation of a kernel bandwidth used for their distance metric which can be computationally expensive. $\text{CAEA}_{\text{Drift}}$ (Takebayashi et al., 2023) is a variant of CAEA that can handle evolving data streams. However, its node deletion mechanism doesn't allow it to retain previously learned distributions when they're no longer presented in the input stream.

In order to summarize the positioning of our work with respect to these numerous works, table 1 indicates for each mentioned model/algorithm of continual learning: if it is able to memorize "previous" contexts (thus avoiding catastrophic forgetting), if it deals with an online stream of data, if it performs unsupervised learning, if the identites of tasks/contexts are detected by the model, and if the distributions underlying each context are slowly drifting.

3 Formalism

Generally speaking, continual learning deals with data samples provided in different kinds of situations. The situation can change over different time scales. Our objective in this section is to give a formal definition of a stream of samples.

The random signal of the samples and their context is defined by the joint variable $\mathbf{K}_t \stackrel{\text{def}}{=} (\mathbf{Z}_t, \mathbf{C}_t)$ t.v.i. $\mathcal{Z} \times \mathcal{C}$ (t.v.i. stands for "taking values in"). Our time index is denoted by t. The random signal $\mathbf{Z}_t$ is the actual stream of samples received for learning and $\mathbf{C}_t$ the contextual information. The context, $\mathbf{C}_t$, captures the determining factors that can change suddenly. It has to be included in the formalism even if, in practical cases, the context may not be explicitly provided. The context sequence $\mathbf{C}_t$ tells the current context of the sample stream. Typically, the stream is in the same context for several consecutive samples, then it switches to another context for next consecutive samples, and so on. This kind of behavior for $\mathbf{C}_t$ is conveniently modeled as the observation from a hidden Markov Model (HMM).

To further specify the formalism, we define $(\mathbf{Z}_{\varphi,c})_{(\varphi,\, c)\in\Phi\times\mathcal{C}}$, $\mathbf{Z}_{\varphi,c}$ t.v.i. $\mathcal{Z}$ a family of random variable. The variable $\varphi \in \Phi$ is for the parameters that slowly change, implementing a drift, as opposed to $c \in \mathcal{C}$ for values of fast changing context parameter. The random signal $\mathbf{K}_t$ is such that:

$$\forall t,\ \exists \varphi,\ \forall c,\ \mathrm{P}_{\mathbf{Z}_t \,|\, \mathbf{C}_t = c} = \mathrm{P}_{\mathbf{Z}_{\varphi,c}} \tag{1}$$

where $\mathrm{P}_{\mathbf{X}}$ denotes the law of the random variable $\mathbf{X}$. In other words, once the context is determined, the random variable modeling the sample distribution only depends on a variable φ. For the task of identifying the context from the samples only, a necessary condition is:

$$\forall \varphi,\ c \neq c' \Rightarrow \mathrm{P}_{\mathbf{Z}_{\varphi,\, c}} \neq \mathrm{P}_{\mathbf{Z}_{\varphi,\, c'}}. \tag{2}$$

Algorithm	**Memorizing contexts**	**online**	**Unsupervised**	**Unknown context**	**Drift**
EWC Kirkpatrick et al. (2017)	yes	no	no	no	no
SI Zenke et al. (2017)	yes	yes	no	no	no
XdG Masse et al. (2018)	yes	no	no	no	no
LwF Li & Hoiem (2017)	yes	no	no	no	no
FROMP Pan et al. (2020)	yes	no	no	no	no
A-GEM Chaudhry et al. (2019)	yes	yes	no	no	no
GRS Kurle et al. (2019)	yes	no	no	no	yes
RSB Korycki & Krawczyk (2021)	yes	yes	no	no	yes
Learn++.NSE Elwell & Polikar (2011)	yes	no	no	no	yes
SOMLP Bashivan et al. (2019)	yes	no	no	no	no
GDM Parisi et al. (2018)	yes	no	no	no	no
Gepperth & Karaoguz Gepperth & Karaoguz (2016; 2017)	yes	no	no	no	no
CIM-based (TCA, CAEA, CAE, CAEAC) Masuyama et al. (2019; 2022; 2026; 2023)	no	yes	yes	no	no
SOINN-based (ASOINN, SOINN+, GSOINN+) Shen & Hasegawa (2008); Wiwatcharakoses & Berrar (2020; 2021)	no	yes	yes	no	no
$\text{CAEA}_{\text{Drift}}$ Takebayashi et al. (2023)	no	yes	yes	no	yes
our model (section 6)	yes	yes	yes	yes	yes

Table 1: Comparison of continual learning methods with emphasis on their ability to retain previously learned contexts over time, along with their online processing, supervision level, handling of unknown contexts, and adaptation to concept drift.

In the next section, we propose a benchmark for continual learning formulated with this formalism. Then, section 5 redefines several continual learning tasks proposed in the litterature with the proposed formalism.

4 A benchmark for evaluating online continual learning

In this section, let us introduce a problem used as a benchmark to evaluate the ability of an algorithm to handle a non stationary contextualized distribution of inputs, such as defined in section 3. In order to quantify the performances, we frame the problem as a stream of samples, where most of the samples are drawn from a distribution of points and few of them are out of distribution (i.e. rare abnormal samples). Among the out of distribution samples, some can be "dangerous" and the others "benign".

4.1 Sample distributions

Following the notations introduced in section 3, we define the samples z_t from a random variable $\mathbf{Z}_t \stackrel{\text{def}}{=} (\mathbf{X}_t, \mathbf{A}_t, \mathbf{D}_t)$ t.v.i. $\mathcal{Z} = \mathbb{R}^2 \times \mathbb{B} \times \mathbb{B}$, where $\mathbb{B} = \{\text{true, false}\}$ and $\mathbf{X}$, $\mathbf{A}$ and $\mathbf{D}$ denote respectively the points, their abnormality, their dangerousness. Therefore, we distinguish four cases:

- $(x, \text{false}, \text{true})$ would be a normal and dangerous point. No such data points exist in the benchmark,

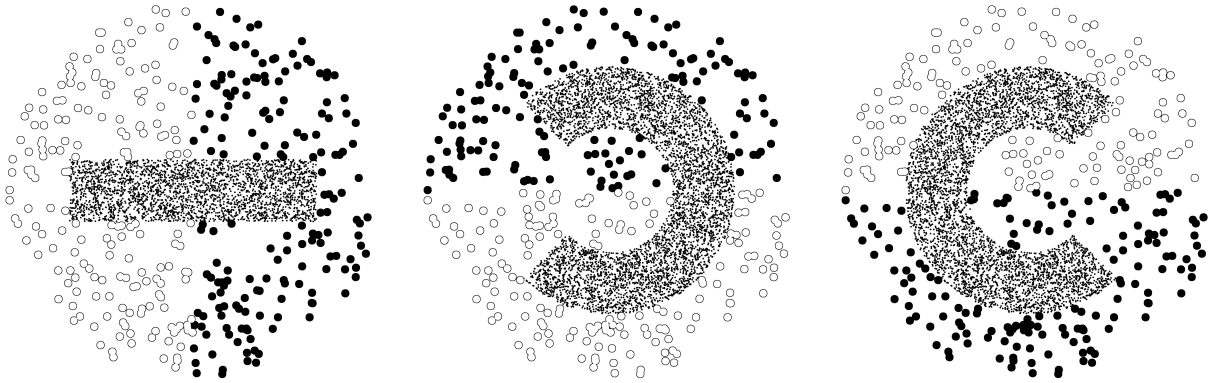

Figure 1: Samples z_t for each of the three contexts, with $\varphi = 0$. Small dots correspond to the many normal (and thus non dangerous) samples $(x, \text{false}, \text{false})$, filled circles to abnormal non dangerous samples $(x, \text{true}, \text{false})$ and empty circles to abnormal dangerous samples $(x, \text{true}, \text{true})$. The disk of noisy samples has a diameter of 3. The ratio of the density in noisy areas over the density of normal areas is 0.02.

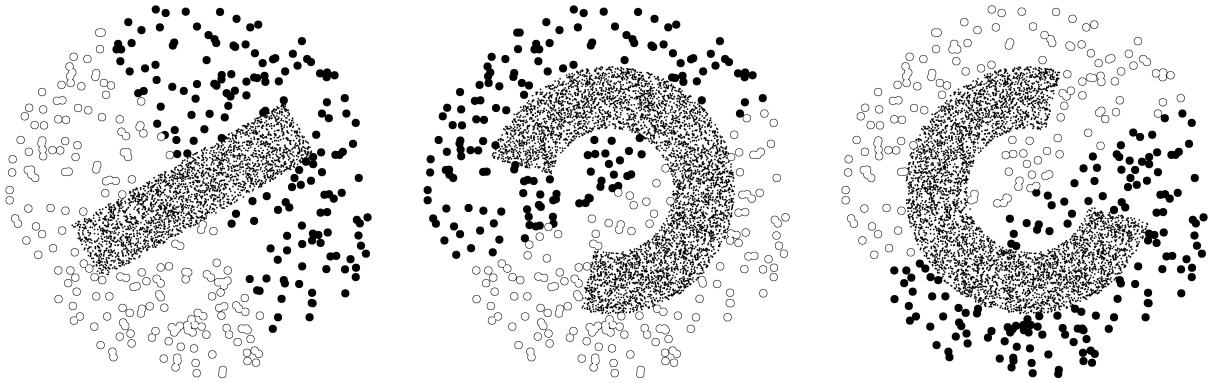

Figure 2: Samples z_t for each of the three contexts, with $\varphi = \pi/6$.

- $(x, \text{false}, \text{false})$ is a normal and non dangerous point, which is the most frequent case in our benchmark,
- $(x, \text{true}, \text{false})$ is an abnormal but non dangerous point, which is the case for half of the abnormal points in our benchmark,
- $(x, \text{true}, \text{true})$ is an abnormal and dangerous point, which is the case for the other half of the abnormal points in our benchmark.

We consider the stream of input samples determined by three different contexts modelled by the random variable $\mathbf{C}_t$ t.v.i. $\mathcal{C} = \{1,\ 2,\ 3\}$. As introduced in section 3, $\mathbf{Z}_t$ and $\mathbf{C}_t$ are viewed as the marginal variables of the joint $\mathbf{K}_t$ random signal driving our input process. The formalism introduces the $\mathbf{Z}_{\varphi,c}$ random variables. Here, obviously, $c \in \{1,\ 2,\ 3\}$. For the φ parameter, let us use an angular value $\varphi \in [0,\ 2\pi[$. Figure 1 shows $\mathbf{Z}_{0,1},\ \mathbf{Z}_{0,2},\ \mathbf{Z}_{0,3}$. Other $\mathbf{Z}_{\varphi,c}$ are obtained from the $\varphi = 0$ case by a rotation of φ. For example figure 2 shows $\mathbf{Z}_{\pi/6,1},\ \mathbf{Z}_{\pi/6,2},\ \mathbf{Z}_{\pi/6,3}$.

We alternate the three contexts as follows. We toss a random permutation of context, and we present them in that order. Then, we toss another permutation, and present the contexts... and so on. This enables to have all the possible transitions between contexts without taking the risk of having one context missing for a too long period. Presenting a context is done by providing the samples according to that context. The number

of samples provided before switching to the next context is taken randomly in $[50000,\ 55000]$. Modeling this temporal random process properly can be done with a hidden Markov model, as mentionned in section 3. In parallel with the short term context switches, the situation φ is updated by $\varphi_{t+1} = \varphi_t + \eta$, adding at each time step a tiny angle increment η to the whole distributions. By construction, this benchmark satisfies equation 1. Moreover, the necessary condition, ensuring that the context influences the sample distribution (cf. equation 2) is actually met. However, as $\mathbf{Z}_{\varphi,1},\ \mathbf{Z}_{\varphi,2},\ \mathbf{Z}_{\varphi,3}$ strongly overlap, identifying that the stream is subject to context switches, among three contexts, by only considering the stream of $\mathbf{X}_t$ samples is quite challenging.

In each context, the problem of detecting dangerous samples is a very unbalanced non-linear classification task. However, if we restrict to abnormal samples in that context, the problem becomes linearly separable. Hence a strategy for deciding on the dangerousness of a sample is to first recognize the context to decide if the sample is normal or not and then to apply the context dependent linear separator. This design enables to challenge anomaly detection algorithms since, once anomalies are detected, the classification becomes straightforward.

4.2 Evaluation of performance

With the sample distributions defined previously, we can set up two incremental learning tasks : unsupervised anomaly detection and supervised dangerousness classification. For both tasks, we will use a binary loss $\ell(a,\ b) \stackrel{\text{def}}{=} 1 - \delta_{a,b}$.

For the unsupervised anomaly detection, the learning algorithm relies on $z_t = x_t$. The evaluation metric is computed from the predicted and ground truth a_t. The evaluation process is performed online, hence the metric is computed from the algorithm algo_t available at time t. Given a stream of samples, the online process involves three consecutive steps :

$$\begin{array}{lcl} \widehat{a}_t & \stackrel{\text{def}}{=} & \text{predict}(\text{algo}_t,\ x_t) \\ \text{err}_t & \stackrel{\text{def}}{=} & \ell(\widehat{a}_t,\ a_t) \\ \text{algo}_{t+1} & \stackrel{\text{def}}{=} & \text{learn}(\text{algo}_t,\ x_t) \end{array} \tag{3}$$

For the supervised dangerousness classification, the learning algorithm relies on $z_t = (x_t, d_t)$. The evaluation metric is computed online from the predicted and ground truth d_t. Given a stream of samples, the online process involves three consecutive steps :

$$\begin{array}{lcl} \widehat{d}_t & \stackrel{\text{def}}{=} & \text{predict}(\text{algo}_t,\ x_t) \\ \text{err}_t & \stackrel{\text{def}}{=} & \ell\Big(\widehat{d}_t,\ d_t\Big) \\ \text{algo}_{t+1} & \stackrel{\text{def}}{=} & \text{learn}(\text{algo}_t,\ x_t,\ d_t) \end{array} \tag{4}$$

In both tasks, the error signal err_t is used to quantify the performances of the algorithm throughout time.

5 Casting the existing continual learning tasks into the proposed formalism

In this section, we reformulate the three types of incremental learning proposed in van de Ven et al. (2022), namely task incremental (TIL), domain incremental (DIL) and class incremental learning (CIL). In Wang et al. (2024), the authors also distinguish several types of continual learning with 7 categories but their formalism is similar to van de Ven et al. (2022). Indeed, for both papers, the data distributions behind every context (in the terminology of van de Ven et al. (2022)) or task (in the terminology of Wang et al. (2024)) are stationnary since these do not change in time (is not dependent on the batch index, in the terminology of Wang et al. (2024)).

In van de Ven et al. (2022) the authors illustrate these three types of learning with the labeled MNIST digits, which provides a convenient illustration of the continual learning scenarios. To fit into the formalism, we first

denote the random variable of the labeled MNIST samples as $\mathbf{M} \stackrel{\text{def}}{=} (\mathbf{I}, \mathbf{G})$ t.v.i. $\mathbb{R}^{28\times 28} \times \{0, \ldots, 9\}$. The authors consider a within-context label and a global label that we model as additional tags to the variable $\mathbf{M}$:

$$\begin{aligned} &f(i,\ g) \stackrel{\text{def}}{=} (i, g, \text{is\_odd}(g)\,, g \text{ div } 2+1) \\ &\mathbf{M}' \stackrel{\text{def}}{=} f(\mathbf{M}) \stackrel{\text{def}}{=} (\mathbf{I}, \mathbf{G}, \mathbf{L}, \mathbf{C}) \text{ t.v.i. } \mathbb{R}^{28\times 28} \times \{0, \ldots, 9\} \times \mathbb{B} \times \mathcal{C} \end{aligned} \tag{5}$$

From the above definition, the context variable is $\mathbf{C}$ t.v.i. $\{1, \ldots, 5\}$. In line with the global label definition g above and its link with the definition of the context index value, images of digits 0 and 1 are presented in context 1, images of digits 2, and 3 and only presented in context 2, etc. All the three learning problems considered in van de Ven et al. (2022) are supervized learning. Therefore, the sample random variable $\mathbf{Z}$ is a joint input-output variable $(\mathbf{X}, \mathbf{Y})$ which will be specified depending on the problem. For the three problems, the samples are presented in batches for each context with N samples for context 1, then N samples for context 2, ... and so on. In our formalism, this sequence of context c_t is modeled as the observations of a simple Markov chain: $\overbrace{1, \ldots, 1}^{N}, \overbrace{2, \ldots, 2}^{N}, \ldots, \overbrace{5, \ldots, 5}^{N}$. This sequence of contexts is considered by the authors to study the extent to which the learning algorithms are robust to catastrophic forgetting.

In these tasks, there is no drift. Therefore, we introduce a dummy singleton $\Phi \stackrel{\text{def}}{=} \{\varphi_0\}$. The random variable for the stream of inputs is

$$\mathbf{Z}_{\varphi,c} \stackrel{\text{def}}{=} \mathbf{Z}_{\varphi_0,c} \stackrel{\text{def}}{=} (\mathbf{X}, \mathbf{Y}) \mid \mathbf{C} = c \tag{6}$$

Equiped with all these definitions, we can now reformulate the three learning problems.

5.1 Task incremental learning

In this learning problem as defined by van de Ven et al. (2022), the context value is explicitelty provided with the image, both during training and testing, and the expected output is the within-context label : $(\mathbf{X}, \mathbf{Y}) \stackrel{\text{def}}{=} ((\mathbf{I}, \mathbf{C})\,, \mathbf{L})$. Examples of data samples are depicted below:

$$\ldots,\ \left(\left(\boxed{9}, 5\right), \text{true}\right),\ \left(\left(\boxed{8}, 5\right), \text{false}\right),\ \left(\left(\boxed{3}, 2\right), \text{true}\right),\ \ldots \tag{7}$$

When the context changes, new pairs of digits are presented and the task is to classify the parity of the digits. The context provided in the stream allows to remove ambiguities on the digit image. For example, the label of $\boxed{3}$ is ambiguous, it could either be a 3 which is odd or a 8 which is even. The digits 3 are only provided in context 2 and the digits 8 are only provided in context 5. Therefore, the stream may contain $\left(\left(\boxed{3}, 2\right), \text{true}\right)$ or $\left(\left(\boxed{3}, 5\right), \text{false}\right)$ where the label to be predicted is different but the context enables solving this ambiguity.

5.2 Domain incremental learning

In this learning problem as defined by van de Ven et al. (2022), the context is no more explicitly provided with the image and the expected output is still the within-context label: $(\mathbf{X}, \mathbf{Y}) \stackrel{\text{def}}{=} (\mathbf{I}, \mathbf{L})$. Examples of data samples are depicted below:

$$\ldots,\ \left(\boxed{3}, \text{false}\right),\ \left(\boxed{0}, \text{false}\right),\ \left(\boxed{9}, \text{true}\right),\ \ldots \tag{8}$$

In this setting, the densities on the inputs are dependent on the context. Mathematically, this reads $c_0 \neq c_1 \Rightarrow \mathrm{P}_{\mathbf{X}\,|\,\mathbf{C}=c_0} \neq \mathrm{P}_{\mathbf{X}\,|\,\mathbf{C}=c_1}$. To keep illustrating with the digits classification problem, when the context changes, new pairs of digits are presented but the task remains to classify whether the digit is odd or even.

5.3 Class incremental learning

In this learning problem as defined by van de Ven et al. (2022), the problem is the standard MNIST classification task: $(\mathbf{X}, \mathbf{Y}) \stackrel{\text{def}}{=} (\mathbf{I}, \mathbf{G})$. Examples of data samples are depicted below:

$$\ldots, \left(\boxed{}, 9\right), \left(\boxed{}, 5\right), \left(\boxed{}, 3\right), \left(\boxed{}, 3\right), \left(\boxed{}, 7\right), \left(\boxed{}, 8\right), \ldots \tag{9}$$

In this setting, new pairs of digits are still presented when the context changes. However, the label to be predicted changes compared to the two other settings. Now, the task is a 10-way classification of the digit. It is no more just the parity of the digit to classify but the identity of it.

5.4 Beyond incremental learning

All the categories of continual learning identified by van de Ven et al. (2022) and Wang et al. (2024) fit into the proposed formalism of section 3, using a statistical formalization approach comparable to that proposed by Wang et al. (2024). However, formalizing these categories in this way fails to leverage the full expressiveness of the formalism, thus highlighting aspects of continual learning that the most usual categories of continual learning do not address. These are precisely the aspects that our study aims to explore further.

6 Tracking and storing representations of distributions

One statement in our paper is that topology-preserving vector quantization methods possess the necessary properties to address continual learning, as formalized in section 3. This claim is supported by the rationale detailed in this paragraph.

Topology-preserving vector quantization methods enable to set up a graph from a dataset of samples, as introduced by Growing Neural Gas approaches Fritzke (1994). A refinement of this algorithm, named GNG-T, dedicated to the tracking of non-stationary distributions Frezza-Buet (2008; 2014), is used hereafter to illustrate our rationale. Using such an algorithm online, throughout the providing of samples, enables to maintain a graph updated, so that it *represents* the current distribution of samples. Figure 3 illustrates this graph at different time instants, for the three contexts considered in our benchmark. Let us call this graph a *tracker*.

Using the tracker, we can determine two crucial features. First, the tracker can be used to tag incoming samples as abnormal or not. Indeed, as the tracker represents the current distribution, it can be used to detect anomalies. These are defined as samples outside of the distribution represented by the graph. Second, the graph gets updated throughout the providing of input samples. This updates smoothly the vertices positions but, more dramatically, it also adds or remove edges or vertices, reconsidering the topology of the graph. Therefore, when the graph is updated, one can detect that the update is significant. If a lot of recent updates are significant, the tracker is *unstable*, meaning that the stream of samples is not stationary. Having a robust and stable graph structure when the distribution is actually stationary, even when it is noisy, is a key feature of GNG-T.

As figure 3 suggests, the tracker might have distinct graph structures in each context. Such structures can be registered as so-called *configurations*, ideally aiming for one configuration per identified context. Let us denote by Ω the collection of configurations learned so far and by ω one of them. A registered configuration gathers pieces of information, one of them being a graph obtained from the tracker, denoted by $\omega.g$. All the $\omega.g$ registered in Ω reflect the structure of the tracker in each configuration. Recognizing a tracker configuration already seen in the past can be made by comparing the current tracker to the registered $\omega.g$ values, using a distance between graphs. The rationale proposed in this paper and detailed above is summarized in algorithm 1

Some sub-processes will be detailed in the experiment section 7. At this general level, using topology-preserving vector quantization enables to detail the configuration update further. Indeed, the current tracker can be compared to the configurations previoulsy stored. If the closest configuration is very different from the

Algorithm 1 `continual_process`(`Z`)

```
buffer_N                                                    # a N-sized circular sample buffer
Ω = ∅                                                       # the set of registered tracker configurations
g = (V = ∅, E = ∅)                                          # the tracker (no vertices and no edges at start)
while true do
    S ← Z(ΔN)                                               # get ΔN next samples from the source Z
    perf ← measure_perf(Ω, g, S)                            # compute performance scores on new samples (see algorithm 7) for further analysis
    insert(buffer_N, S)                                     # add new samples in the circular buffer
    stable ← tracker_update(g, buffer_N)                    # update the tracker (see algorithm 4)
    if stable then
        configs_update(Ω, g, buffer_N)                      # update the set of configurations (see algorithm 2)
    end if
end while
```

tracker, according to some graph distance δ, then a new configuration is added to the collection ω, initialized to the tracker. Otherwise, the closest configuration is updated with the current data available stored in a circular buffer. This configuration adaptation process is the main idea for handling large timescale changes introduced in sections 3 and 4, as discussed further. This is summarized in algorithm 2.

Algorithm 2 `configs_update`(Ω, g, buffer_N)

```
ω* ← argmin_{ω∈Ω} δ(g, ω.g)                                 # find the registered configuration closest to the tracker
if δ(g, ω*.g) > γ then
    ω* ← make_config(g)                                     # a new configuration needs to be registred (see algorithm 5)...
    Ω ← Ω ∪ {ω*}                                            # ... so it is added
end if
(buffer°_N, buffer!_N) ← detect_anomalies(g, buffer_N)      # see algorithm 3
current_config_update(ω*, buffer°_N, buffer!_N)             # we update the configuration (see algorithm 6)
```

Algorithm 3 `detect_anomalies`(g, S)

```
S! = { z ∈ S, min_{v∈V(g)} ||v.x − z.x|| > ζ }              # identify anomalies
S° = S \ S!                                                 # identify regular samples
return (S°, S!)
```

7 Experiments

In this section, we propose an implementation of the rationale introduced in section 6 on the benchmark presented in section 4. As previously stated, the algorithm used for topologically preserving vector quantization is GNG-T, as detailed in Frezza-Buet (2014). This algorithm relies on data *epochs*, which will be here the data in the circular buffer buffer_N. Therefore, we call *training* a graph the providing of all the samples positions $\{z.\mathtt{x} \mid z \in \text{buffer}_N\}$ to a GNG-T update rule. For our illustration, the most interesting feature of GNG-T is a self-regulation mechanism for adjusting continuously the number of vertices in the graph. It is controlled by the *target* parameter T.

7.1 Implementation of the tracker

The tracker is implemented from a GNG-T process. The handling of noise is implemented as follows. During training, each sample is associated to the closest node in the graph g, as most vector quantization techniques do. So during training, each vertex $v \in \mathcal{V}(g)$ can compute the average distance $v.\mathtt{var}$ as follows:

$$
\begin{aligned}
v.\texttt{var} &\stackrel{\text{def}}{=} \frac{\displaystyle\sum_{z \in \text{closest}(v)} \|v.\texttt{x} - z.\texttt{x}\|^2}{|\text{closest}(v)|} \\
\text{where} \quad \text{closest}(v) &\stackrel{\text{def}}{=} \left\{ z \in \text{buffer}_N \;\middle|\; v = \operatorname*{argmin}_{v' \in \mathcal{V}(g)} \|v'.\texttt{x} - z.\texttt{x}\| \right\}
\end{aligned}
$$

The vertices for which $v.\texttt{var} > \epsilon$ are considered as representing noisy regions. Indeed, vector quantization is sensitive to input density, vertices in low-density regions tend to be sparse. These vertices are said *insensitive*, as well as the edges connecting them. Figure 3 illustrates how GNG-T enables to deploy a graph, used as a tracker, on the data provided by our benchmark. On this figure, the "insensitive" vertices and edges are drawn in light gray.

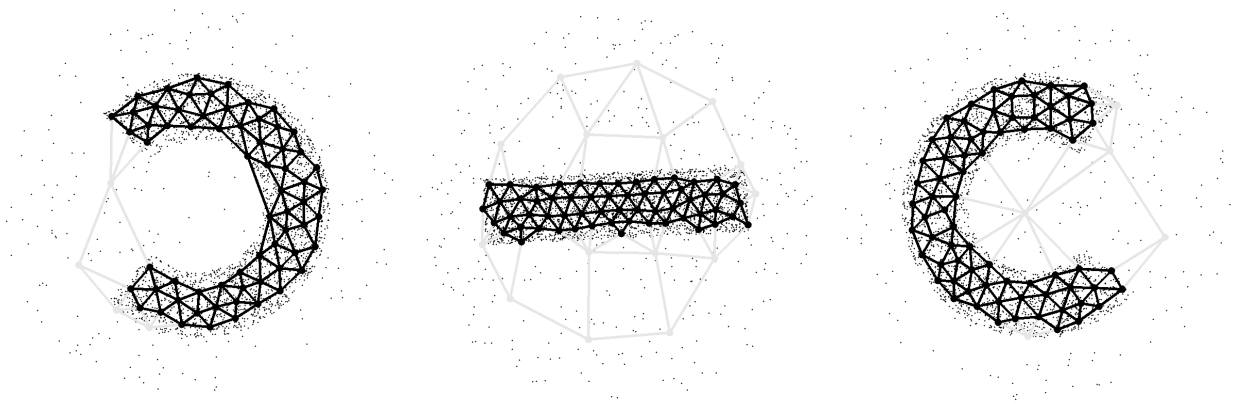

Figure 3: Tracking the non stationary distribution with GNG-T. Snapshots are taken in each context. Gray edges and vertices are the ones referred to as *insensitive*, i.e. identified as representing noisy regions.

In the benchmark, data is $z \stackrel{\text{def}}{=} (z.\texttt{x}, z.\texttt{a}, z.\texttt{d}) \in \mathbb{R}^2 \times \mathbb{B} \times \mathbb{B}$, i.e. the position, the abnormality tag, the dangerousness tag. Only the position $z.\texttt{x}$ is used to compute graphs, the abnormality tag $z.\texttt{a}$ is only used to measure performances. The dangerousness tag $z.\texttt{d}$ is used to train the dangerousness detector. Of course, to prevent data contamination in the evaluation process, as introduced in section 4.2, the performance of the detector is measured on data samples before these are used for training (test-then-train approach).

The similarity measure of graphs, in line 2 of algorithm 2, is very simple in the present experiments. It only considers vertices and is defined as :

$$
\delta'(g_1,\ g_2) \stackrel{\text{def}}{=} \begin{cases} \displaystyle\sum_{v_2 \in \mathcal{V}(g_2)} \min_{v_1 \in \mathcal{V}(g_1)} \|v_1.\texttt{x} - v_2.\texttt{x}\|^2 & \text{if } |\mathcal{V}(g_1)| \times |\mathcal{V}(g_2)| \neq 0 \\ +\infty & \text{otherwise} \end{cases}
$$

$$
\delta(g_1,\ g_2) \stackrel{\text{def}}{=} \sqrt{\min\left(\frac{\delta'(g_1,\ g_2)}{|\mathcal{V}(g_2)|}, \frac{\delta'(g_2,\ g_1)}{|\mathcal{V}(g_1)|} \right)}
$$

The same distance is used for detecting the stationarity of the input stream as returned by the `tracker_update` function called in line 8 of algorithm 1 and defined by algorithm 4.

7.2 Registered configurations

The rationale introduced in this paper is based on configurations that are registered and updated. A configuration is to be somehow “recognized” from the input stream. It is associated to a context, in the

Algorithm 4 `tracker_update`$(g,\ \text{buffer}_N)$

$g' \leftarrow g$ # we save the current tracker state.
`process_gngt_epoch`$(g,\ \text{buffer}_N,\ \epsilon)$ # we update the tracker, detecting noisy vertices with ϵ threshold.
return $\delta(g,\ g') < \sigma$ # we return stability status.

meaning of contexts introduced in section 3. With the benchmark used in these experiments, there are three contexts, illustrated in figures 1 and 2. We have set up the benchmark such as recognizing the current context actually helps. Indeed, once the context is recognized, one can apply a simple linear separation to distinguish between dangerous and non-dangerous events. The linear separability constraint is not met when all the samples in buffer_N are considered, but it is obtained when only abnormal samples $\text{buffer}^!_N$ are taken into account. Once the anomalies are detected, a simple single-layer perceptron is sufficient to classify dangerousness.

For these reasons, a configuration $\omega \stackrel{\text{def}}{=} (\omega.g, \omega.\texttt{sep})$ is a graph, representing the non-anomaly sample distribution, as well as a linear separator, enabling to detect dangerous samples among abnormal samples. Let us denote the linear separators as $[w,\ b] \stackrel{\text{def}}{=} x \mapsto \text{sign}\Big(w^{\mathsf{T}}x + b\Big)$. The setting up of a new configuration, called in line 3 of algorithm 2 is detailed in algorithm 5. The update of an existing configuration, called in line 7 of algorithm 2 is detailed in algorithm 6.

Algorithm 5 `make_config`(g)

ω # define a new configuration
$\omega.\texttt{sep} \leftarrow [0,\ 0]$ # we initialize the linear classifier
$\omega.g \leftarrow$ a copy of sensitive vertices and sensitive edges of g only.
return ω

Algorithm 6 `current_config_update`$\big(\omega,\ S^\circ,\ S^!\big)$

$S^\circ,\ S^!$ # normal and abnormal input samples
$i \leftarrow 1$ # the following code updates $\omega.g$
repeat
  is_stable $\leftarrow$ `process_gngt_epoch`$(\omega.g,\ S^\circ)$ # this is an epoch-pass on non-anomaly samples, it returns true if no vertices and no edges have been modified.
  $i \mathrel{+}= 1$
until is_stable $\vee\ i = E_{\max}$
$i \leftarrow 1$ # the following code updates $\omega.\texttt{sep}$
repeat
  pick a sample $(z.\texttt{x}, z.\texttt{d})$ in $S^!$, $i \mathrel{+}= 1$ # we get a position/dangerousness pair from anomalies
  $[w,\ b] = \omega.\texttt{sep}$
  if $z.\texttt{d}$, $y \leftarrow 1$, else $y \leftarrow -1$.
  if $y\Big(w^{\mathsf{T}}z.\texttt{x} + b\Big) < 0$ **then**
    $\omega.\texttt{sep} \mathrel{+}= [\alpha y x,\ \alpha y]$ # perceptron learning rule
  end if
until $i = K$

7.3 Experimental setup

In order to evaluate our approach, we use the benchmark introduced in section 4. As mentionned in section 4, at each sampling, the φ angle of rotation is incremented by $\eta = 3 \times 10^{-5}$ degrees, which is a slow change compared to the context transitions time scales.

In our experiments, we refill the circular sample buffer by bunches of ΔN samples (see line 5 in algoritm 1). We then measure the performance at each refill and plot it graphically (see line 6 in algorithm 1). As the process is continual, the experiments consist of a constant stream of refills, thus providing a continual flow of measurements. As the data is artificial, we know which refills contain a context transition. We record the performance of such a transition refill, along with that of the next 89 refills, in a "performance line" in order

to obtain a performance line of length 90. We wait for the next context transition refill to start the next performance line.

Figures 5, 6, 7, show registered configurations at various performance line numbers. Performance measures are reported, performance line by performance line, in figure 4. They have been obtained by algorithm 7 for both anomaly and dangerousness detection.

Algorithm 7 `measure_perf`$(\Omega,\ g,\ S)$

$\omega^\star \leftarrow \underset{\omega\in\Omega}{\operatorname{argmin}}\, \delta(g,\ \omega.g)$ # find the registered configuration closest to the tracker
$\left(S^\circ, S^!\right) \leftarrow$ `detect_anomalies`$(\omega^\star.g,\ S)$
for $(z.\mathtt{x}, z.\mathtt{a}, z.\mathtt{d}) \in S^\circ$ **do**
  $\left(z.\widehat{\mathtt{a}}, z.\widehat{\mathtt{d}}\right) \leftarrow (\text{false}, \text{false})$ # we add predictions to non-anomaly samples
end for
for $(z.\mathtt{x}, z.\mathtt{a}, z.\mathtt{d}) \in S^!$ **do**
  $\left(z.\widehat{\mathtt{a}}, z.\widehat{\mathtt{d}}\right) \leftarrow (\text{true}, \omega^\star.\mathtt{sep}(z.\mathtt{x}))$ # we add predictions to anomaly samples, dangerousness is computed from linear classifier of the closest registered configuration
end for
anomaly_score $\leftarrow \text{fscore}\left(\left\{\left(z.\mathtt{a}, z.\widehat{\mathtt{a}}\right) \,\middle|\, z \in S\right\}\right)$ # we compute f-score for anomaly detection...
anomaly_score $\leftarrow \text{fscore}\left(\left\{\left(z.\mathtt{d}, z.\widehat{\mathtt{d}}\right) \,\middle|\, z \in S\right\}\right)$ # ... and f-score for dangerousness detection
return (anomaly_score, dangerosity_score)

The result show that GNG-T enables to track the current context and to provide continuously an anomaly detector. All the three contexts correspond to distributions that overlap, so detecting a change of context cannot be realized on the basis of a single sample position. A computational representation of the current distribution *as a whole* is thus required. This is what a topology-preserving vector quantization algorithm as GNG-T offers.

Once the context is detected, we can do more than only detect anomalies. We can provide something associated to the detected context. Here, this is a linear separator, already trained from past occurrences of this context. As shown by comparing the right middle and right bottom plots in figure 4, this is much more efficient than continuoulsy training a separator on detected anomalies. Context detection actually helps here.

Last, the snapshots of configurations taken throughout the run of experiments (see red dotted lines in figure 4 and corresponding figures 5, 6, 7) show that in spite of a slow continual rotation of the distribution, the configuration set keeps on representing three contexts, which are each smoothly adapted to the rotation. A video of the whole process is available[1]. Thanks to the interplay of the graph similarity measure, the tracker and configuration updates, the proposed algorithm is able to balance between updating the registered configurations and registering a new one while the underlying distributions are drifting.

$N = 3000$	Circular buffer size.
$\Delta N = 500$	Number of new samples acquired at each step.
$T = 0.001$	Target parameter of GNG-T.
$\eta = 3 \times 10^{-5}$	Situation angular change for each sample, in degrees.
$\sigma = 0.06$	Minimum distance between two successive trackers configurations for considering stability.
$\gamma = 0.1$	Maximal graph distance for associating a already registered configuration to the tracker, thus not creating a new configuration.
$\zeta = 0.2$	Distance threshold for anomaly detection.
$\epsilon = 0.14$	Maximal $v.\mathtt{var}$ value for out-of-noise vertices.
$E_{\max} = 100$	Maximal number of epochs allowed for reaching GNG-T graph stability.
$K = 500$	Number of training sample provided to the perceptron for a configuration update.
$\alpha = 0.0005$	Perceptron learning rate.

Table 2: Parameter settings for experiments.

[1] `[ URL OMITTED FOR BLIND REVIEW ]`

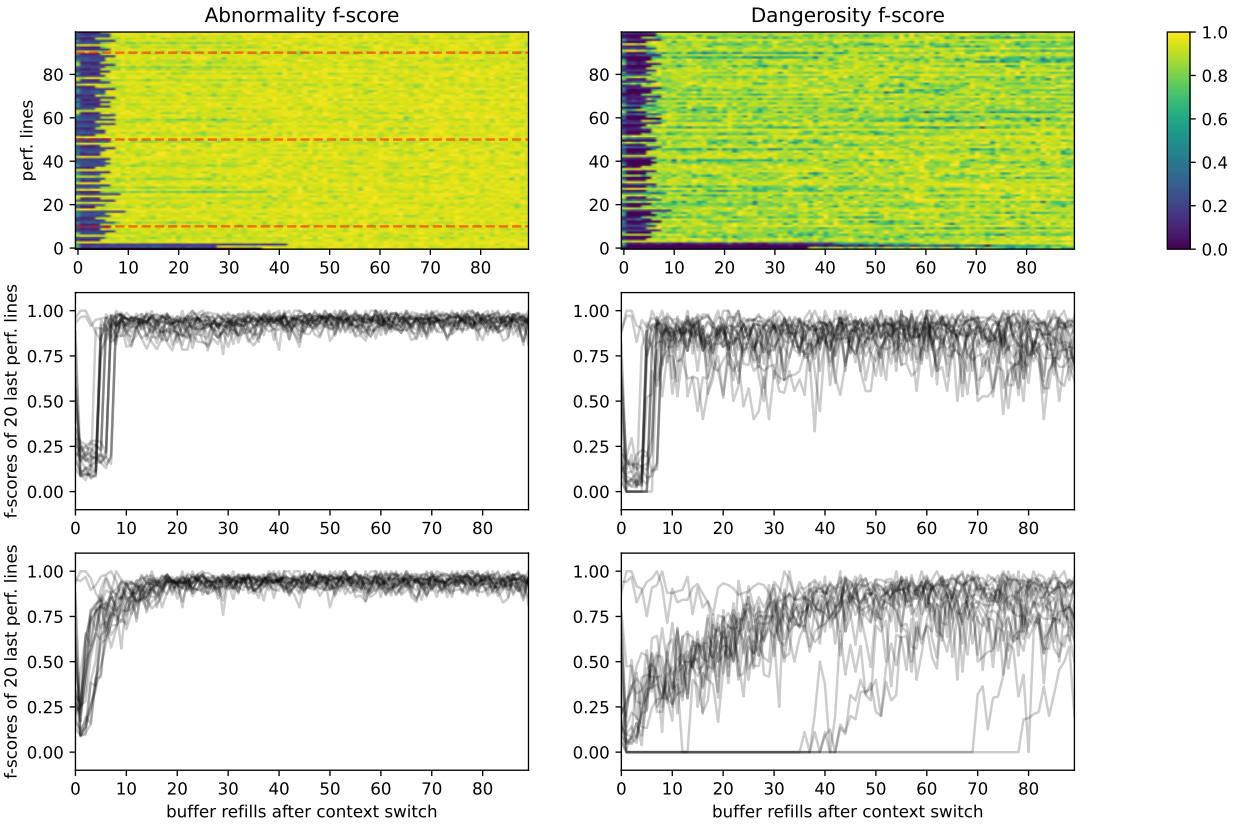

Figure 4: The plots show performance lines. On the left, plots concern anomaly detections, on the right, dangerousness detection. The top plots show all the lines, revealing that performances decrease at context switches for a short while, before recovery. The red dotted lines notify the performance lines for which we have provided the figures for registered configurations. The middle line plots shows the 20 last performance lines. For the two bottom line plots, the configuration set is restricted to a single element. This exhibits the advantage of registering configurations.

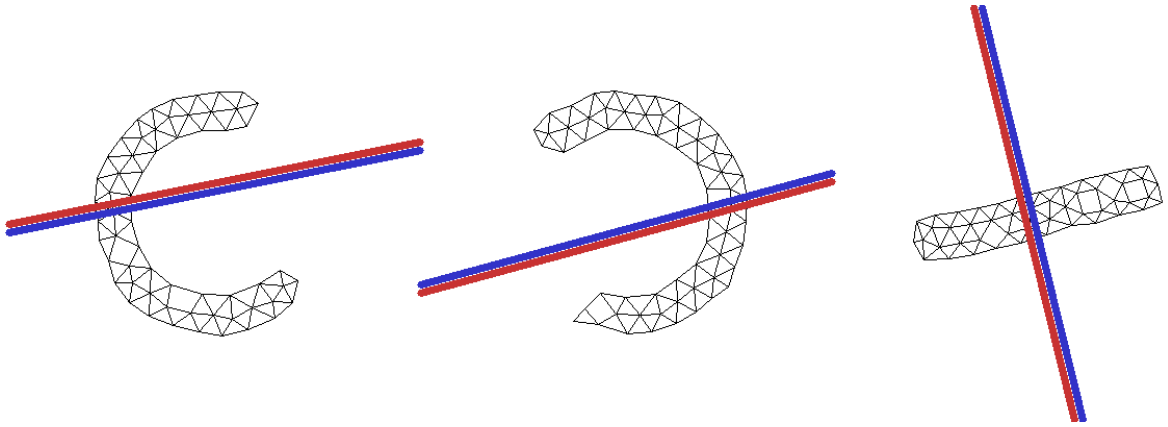

Figure 5: Registered configurations at performance line number 10. The separator is depicted as a double colored line, the red line is on the positive side.

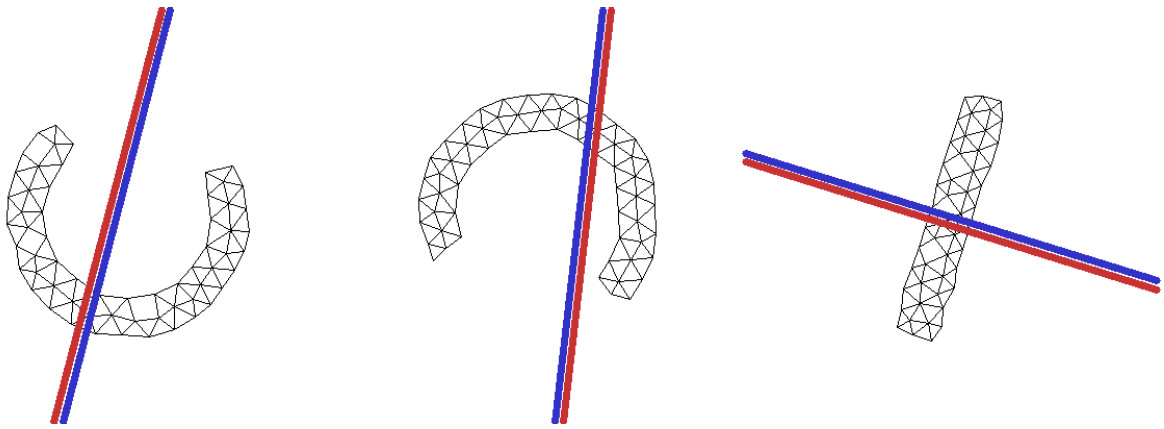

Figure 6: Registered configurations at performance line number 50. The separator is depicted as a double colored line, the red line is on the positive side.

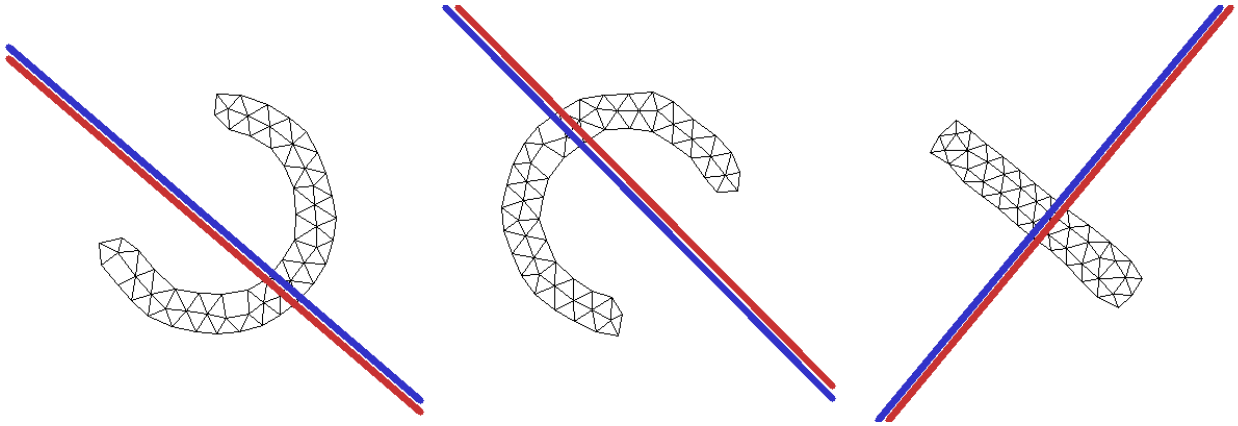

Figure 7: Registered configurations at performance line number 90. The separator is depicted as a double colored line, the red line is on the positive side.

8 Conclusion

Continual learning is commonly motivated by the need to extend the capabilities of an already trained model when new data become available. In this setting, a central challenge is catastrophic forgetting: the objective is not merely to use a pre-trained model as a starting point for learning a new task, but rather to acquire new knowledge while preserving previously learned representations. Framing this problem within the continual learning paradigm naturally places it in the context of non-stationary data streams. Indeed, sequentially learning from one dataset and then from another one can be interpreted as a particular form of continual learning in which the underlying data distribution abruptly shifts from one regime to another. However, such a scenario represents only a narrow subset of the broader class of continual learning schemes. For this reason, we advocate for a more general perspective on continual learning.

To address this broader setting, we adopt a twofold approach. First, we introduce a formal framework based on random signals that encompasses both existing formulations and more general non-stationary scenarios. From this framework, we derive an online evaluation protocol suitable for continual learning systems. Second, we propose a benchmark consisting of an artificial non-stationary data stream designed to make context detection operationally meaningful in the context of danger detection. In this benchmark, identifying the current context allows abnormal samples to be distinguished from normal ones and, subsequently, enables a linear classifier to determine whether these abnormal samples should be considered dangerous. The performance of an algorithm in predicting the dangerousness of incoming samples has to be assessed online, in accordance with the scope of continual data processing. Furthermore, the benchmark supports the introduction of gradual distributional drifts, allowing the underlying data-generating process to evolve continuously over time. Robustness to such drifts remains relatively unexplored in the continual learning literature.

Having outlined a broader conception of continual learning, we argue that topology-preserving vector quantization methods constitute promising candidates for addressing these challenges. These techniques continuously adapt their internal graph structures as new data are observed, while simultaneously providing an explicit representation that can support context identification and recognition. By maintaining such graph-based representations, topology-preserving vector quantization offers a natural mechanism to handle a non stationary contextualized distribution of inputs. To illustrate this perspective, we evaluate the GNG-T algorithm on the proposed benchmark and introduce a methodology for conducting analyses and experiments within a continual learning framework.

Many studies in continual learning motivate the field through the example of an autonomous agent operating in a continuously changing environment, yet subsequently focus on the narrower problem of updating a pre-trained model with new data while avoiding forgetting. In contrast, the present work goes beyond this restricted viewpoint, although it remains grounded in experiments on artificial data. An important direction for future research is the identification of realistic application domains in which the discovery of hidden contexts would improve the performance of the task. Demonstrating the usefulness of topology-preserving vector quantization techniques as dynamic and adaptive representations in such settings would further highlight their relevance for genuinely continual learning problems.

Acknowledgements

This work was supported by the projet [**OMITTED FOR BLIND REVIEW**].

References

Pouya Bashivan, Martin Schrimpf, Robert Ajemian, Irina Rish, Matthew Riemer, and Yuhai Tu. Continual learning with self-organizing maps. *arXiv preprint arXiv:1904.09330*, 2019.

Rakesh Chalasani and Jose C Principe. Self-organizing maps with information theoretic learning. *Neurocomputing*, 147:3–14, 2015.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-GEM. In *International Conference on Learning Representations*, 2019.

Ryan Elwell and Robi Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE transactions on neural networks*, 22(10):1517–1531, 2011.

Hervé Frezza-Buet. Following non-stationary distributions by controlling the vector quantization accuracy of a growing neural gas network. *Neurocomputing*, 71(7-9):1191–1202, March 2008.

Hervé Frezza-Buet. Online computing of non-stationary distributions velocity fields by an accuracy controlled growing neural gas. *Neural Networks*, 60:203–221, 2014. ISSN 0893-6080.

Bernd Fritzke. A growing neural gas network learns topologies. In G. Tesauro, D. Touretzky, and T. Leen (eds.), *Advances in Neural Information Processing Systems*, volume 7. MIT Press, 1994.

Alexander Gepperth and Cem Karaoguz. A bio-inspired incremental learning architecture for applied perceptual problems. *Cognitive Computation*, 8(5):924–934, 2016.

Alexander Gepperth and Cem Karaoguz. Incremental learning with self-organizing maps. In *2017 12th International Workshop on Self-Organizing Maps and Learning Vector Quantization, Clustering and Data Visualization (WSOM)*, pp. 1–8. IEEE, 2017.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Lukasz Korycki and Bartosz Krawczyk. Class-incremental experience replay for continual learning under concept drift. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3649–3658, 2021.

Richard Kurle, Botond Cseke, Alexej Klushyn, Patrick Van Der Smagt, and Stephan Günnemann. Continual learning with bayesian neural networks for non-stationary data. In *International Conference on Learning Representations*, 2019.

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

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

Nicolas Y Masse, Gregory D Grant, and David J Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115(44): E10467–E10475, 2018.

Naoki Masuyama, Chu Kiong Loo, Hisao Ishibuchi, Naoyuki Kubota, Yusuke Nojima, and Yiping Liu. Topological clustering via adaptive resonance theory with information theoretic learning. *IEEE Access*, 7: 76920–76936, 2019.

Naoki Masuyama, Narito Amako, Yuna Yamada, Yusuke Nojima, and Hisao Ishibuchi. Adaptive resonance theory-based topological clustering with a divisive hierarchical structure capable of continual learning. *IEEE Access*, 10:68042–68056, 2022.

Naoki Masuyama, Yusuke Nojima, Farhan Dawood, and Zongying Liu. Class-wise classifier design capable of continual learning using adaptive resonance theory-based topological clustering. *Applied Sciences*, 13 (21):11980, 2023.

Naoki Masuyama, Takanori Takebayashi, Yusuke Nojima, Chu Kiong Loo, Hisao Ishibuchi, and Stefan Wermter. A parameter-free adaptive resonance theory-based topological clustering algorithm capable of continual learning. *Neural Computing and Applications*, 38(7):217, 2026.

Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard Turner, and Mohammad Emtiyaz E Khan. Continual deep learning by functional regularisation of memorable past. *Advances in neural information processing systems*, 33:4453–4464, 2020.

German I Parisi, Jun Tani, Cornelius Weber, and Stefan Wermter. Lifelong learning of human actions with deep neural network self-organization. *Neural Networks*, 96:137–149, 2017.

German I Parisi, Jun Tani, Cornelius Weber, and Stefan Wermter. Lifelong learning of spatiotemporal representations with dual-memory recurrent self-organization. *Frontiers in neurorobotics*, 12:78, 2018.

German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Furao Shen and Osamu Hasegawa. A fast nearest neighbor classifier based on self-organizing incremental neural network. *Neural networks*, 21(10):1537–1547, 2008.

Takanori Takebayashi, Naoki Masuyama, and Yusukeusuke Nijima. Adaptive resonance theory-based topological clustering with node deletion mechanism for evolving stream data. In *2023 International Conference on Machine Learning and Cybernetics (ICMLC)*, pp. 576–581. IEEE, 2023.

Gido M. van de Ven, Tinne Tuytelaars, and Andreas S. Tolias. Three types of incremental learning. *Nature machine intelligence*, 4(12):1185–1197, 2022. ISSN 2522-5839.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46 (8):5362–5383, 2024.

Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine learning*, 23:69–101, 1996.

Chayut Wiwatcharakoses and Daniel Berrar. Soinn+, a self-organizing incremental neural network for unsupervised learning from noisy data streams. *Expert Systems with Applications*, 143:113069, 2020.

Chayut Wiwatcharakoses and Daniel Berrar. A self-organizing incremental neural network for continual supervised learning. *Expert Systems with Applications*, 185:115662, 2021.

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