Statistical process monitoring of artificial neural networks

Anna Malinovskaya

Leibniz University Hannover, Institute of Cartography and Geoinformatics, Germany

Pavlo Mozharovskyi

LTCI, Télécom Paris, Institut Polytechnique de Paris, France

Philipp Otto

Leibniz University Hannover, Institute of Cartography and Geoinformatics, Germany

July 28, 2023

Abstract

The rapid advancement of models based on artificial intelligence demands innovative monitoring techniques which can operate in real time with low computational costs. In machine learning, especially if we consider artificial neural networks (ANNs), the models are often trained in a supervised manner. Consequently, the learned relationship between the input and the output must remain valid during the model's deployment. If this stationarity assumption holds, we can conclude that the ANN provides accurate predictions. Otherwise, the retraining or rebuilding of the model is required. We propose considering the latent feature representation of the data (called "embedding") generated by the ANN to determine the time when the data stream starts being non-stationary. In particular, we monitor embeddings by applying multivariate control charts based on the data depth calculation and normalized ranks. The performance of the introduced method is compared with benchmark approaches for various ANN architectures and different underlying data formats.

Keywords: Change Point Detection, Data Depth, Latent Feature Representation, Multivariate Statistical Process Monitoring, Artificial Neural Networks, Online Process Monitoring.

1 Introduction

The rapid advancement of Artificial Intelligence (AI) has encouraged practitioners in various scientific fields to examine the benefits and challenges of Machine Learning (ML) algorithms in their research. For instance, in astronomy, the separation of astrophysical objects such as stars and galaxies can be performed by applying decision trees (Ball et al., 2006); reliable forecasts of energy consumption using support vector machines are extensively studied in civil engineering (Gao et al., 2019); in biology, artificial neural networks (ANNs) are applied for predicting molecular traits (Angermueller et al., 2016). Due to the emergence of big data and the increased capability of computers, the development of ANNs has received particularly broad attention (cf. Chiroma et al., 2018). Current ANNs, consisting of many hidden layers and being called "deep learning" models, have achieved impressive results in many areas, such as finance, linguistics, or photogrammetry (cf. Aldridge and Avellaneda, 2019; Otter et al., 2020; Heipke and Rottensteiner, 2020).

Considering supervised learning for tasks such as regression or classification, data with a known relationship between the input and the output is required. During training, the algorithm minimizes the error between the model's output and the target. After the parameter estimation and the testing phase, the model is deployed on new data, promising a stable performance. However, if the distribution of input data changes, it would violate the stationarity assumption and adversely impact predictive performance (see also Huang et al., 2011).

In computer science, "concept drift" refers to the problem of changes in the data distribution that render the current prediction model inaccurate or obsolete (Demšar and Bosnić, 2018). In statistics, "nonstationarity" describes time series with changing statistical properties. Additionally, "novelty" can occur, e.g., when the data stream introduces instances of a new class, being not present during training (cf. Masud et al., 2009; Garcia et al., 2019). Model revision is necessary to address these issues, ranging from retraining with new data to algorithm replacement. Strategies handling nonstationarity without explicit detection exist (cf. Gama et al., 2014), but current research highlights the benefits of employing drift detection methods during model deployment (cf. Piano et al., 2022; Zhang et al., 2023). This minimizes redundant adjustments while maintaining prediction accuracy.

From a statistical point of view, for real-time detection of changes, we can apply online monitoring techniques, e.g., univariate or multivariate control charts (cf. Celano et al., 2013; Psarakis, 2015; Perdikis and Psarakis, 2019). These monitoring methods are primary techniques in Statistical Process

Monitoring (SPM), testing for unusual variability over time (Montgomery, 2020). The combination of control charts and ML algorithms has become a prominent concept for improving SPM procedures (cf. Psarakis, 2011; Weese et al., 2016; Lee et al., 2019; Apsemidis et al., 2020; Zan et al., 2020; Sergin and Yan, 2021; Yeganeh et al., 2022). However, the authors are not aware of competitively using control charts in reverse to supervise the quality of ANN applications in a realistic setting, e.g., considering the limited availability of labels during model deployment and various ANN architectures. Consequently, statistical monitoring of ANN applications is a field offering new perspectives for SPM.

In an ANN, data flows through hidden layers, each containing neurons that perform nonlinear transformations, ultimately producing predictions in the output layer (Hermans and Schrauwen, 2013). The hidden layers combine learned data representations from previous layers, abstracting irrelevant details and extracting useful features for generating the output. Neurons within the ANN, as demonstrated by Wang et al. (2019), possess two crucial properties: stability and distinctiveness. For smooth classification surfaces or prediction functions, nearby samples should activate similar neurons, leading to similar output values. However, in the presence of nonstationary data, such as from an unknown class, neuron values may deviate beyond the typical activation range. In this case, its interim representation generated by ANN before the output layer would be different compared to the interim representation of the data that correctly belongs to a predicted class. Commonly represented by a low-dimensional vector, the learned latent feature representation also called "embedding" comprises a dense summary of the incoming sample. Alternatively, sliced-inverse regression can reduce dimensionality without specifying a regression model (Li, 1991). It projects predictors onto a subspace while preserving information about the conditional distribution of the response variable(s) given the predictors. Various methods have been proposed that utilize the inverse relationship between the variables (Cook and Ni, 2005; Wang and Xia, 2008; Wu, 2008), or sparse techniques facilitating the interpretation (Li and Nachtsheim, 2006; Li, 2007; Lin et al., 2019). Both ways of reducing the dimensionality and compressing the knowledge about a data sample are suitable for SPM.

The majority of multivariate SPM methods are based on the assumption that the observed process follows a specific distribution. In the general case of ANNs, however, the distribution of embeddings is unknown. Although the network could be trained in a way such that the embeddings follow a certain distribution, this concept is too restrictive in practice. In this work, we focus on developing a nonparametric SPM approach using a data depth-based control chart, making no assumptions about the ANN type and the distribution of embeddings.

In Section 2, we review the relevant literature and describe the notation related to ANNs, followed by the definition of the studied change point detection problem. Afterward, we discuss the notion of data depth and respective control charts in Section 3. The experimental results of our approach and comparison to the benchmark methods on both synthetic and real data are provided in Section 4, followed by the discussion about open research directions in Section 5.

2 Background and Notation

Detecting nonstationarity can be explored from different perspectives. In statistics, we would usually look at change point detection methods (cf. Ali et al., 2016), while in computer science, these techniques are rather known as concept drift, anomaly, or out-of-distribution detection (cf. Žliobaitė et al., 2016; Yang et al., 2022b; Fang et al., 2022). In the following sections, we briefly review the relevant literature from different fields (Section 2.1) and introduce important notation of ANNs (Section 2.2) together with the change point detection framework (Section 2.3).

2.1 Literature Review

In general, approaches to detect nonstationarity in ML applications can be subdivided into two groups: performance-based methods and distribution-based methods (Hu et al., 2020). The first group relies on labeled instances, monitoring the classification error rate or other performance metrics (cf. Klinkenberg and Renz, 1998; Klinkenberg and Joachims, 2000; Nishida and Yamauchi, 2007), for example, by conducting sequential hypothesis tests based on the ideas of control charts (cf. Gama et al., 2004; Baena-Garcıa et al., 2006; Kuncheva, 2009; Mejri et al., 2017). Mejri et al. (2021) developed a two-stage time-adjusting control chart for monitoring misclassification rates, which updates the control limits in the first stage, and validates the detected changes in the second stage. However, in practice, labeled data is usually not available when ANNs are applied. Thus, the idea of adapting performance-based approaches to a confidence-related output produced by a model was proposed (Haque et al., 2016; Kim and Park, 2017). If a classifier processes the anomalous data, it is expected that the model's confidence about the affinity of a data point to a certain class would change so that the unusual behavior could be detected (cf. Hendrycks and Gimpel, 2016).

Anomaly detection in distribution-based methods involves the analysis of metrics related to the distribution. A considerable number of techniques apply a sliding window approach on either a one-

4

dimensional data stream or on several features individually, comparing the data distribution of the current window to the reference sample obtained from the training dataset (cf. Bifet and Gavalda, 2007; Bifet et al., 2018; Gemaque et al., 2020). However, the underlying reason related to the notable performance of ANNs is their generalization ability in complex high-dimensional AI tasks such as object or speech recognition (Goodfellow et al., 2016), meaning that the detection of nonstationarity from the original data would not be appropriate due to the excessive number of features. Considering novelty detection, it can be based on parametric density estimates (e.g., using Gaussian mixture models (Roberts and Tarassenko, 1994) or hidden Markov models (Yeung and Ding, 2003)), and non-parametric estimates (e.g., based on k-nearest neighbors (Guttormsson et al., 1999), kernel density estimates (Yeung and Chow, 2002) or string matching (Forrest et al., 1994)). These methods usually require heuristically chosen thresholds to decide about the novelty. The interested reader may refer to Markou and Singh (2003a,b) for a more detailed review.

By considering a latent representation of the original data stream such as embeddings generated by ANNs, outlier and anomaly detection methods based on distance metrics and nearest neighbor approaches were shown to be suitable and efficient in detecting nonstationary samples (cf. Lee et al., 2018; Sun et al., 2022). Particularly beneficial is their capability of providing an overall outlying score that would consider all data features together, leading to a more explicit decision about the observed abnormalities.

Alternatively, other ML algorithms for drift detection such as Support Vector Machine (SVM) (Krawczyk and Woźniak, 2015) or autoencoders as specific types of ANNs (Pidhorskyi et al., 2018) can be applied for change detection. Another suggestion is to use ensemble models consisting of, for instance, several decision trees with a majority voting mechanism (Li et al., 2015). In particular cases, the detection methods can be enhanced through large-scale pre-training techniques, leading to remarkably informative embeddings (Fort et al., 2021). However, in our work, we relax any assumptions on the availability of additional data or specific model architectures. Thus, we describe ANNs from a general perspective in the next section.

2.2 Artificial Neural Networks

The main goal of training ANN is to estimate the parameters ϑ of a highly nonlinear function $f(\cdot, \vartheta)$: $\mathbb{R}^d \to \mathbb{R}^v$, mapping a d-dimensional input (i.e., observed data) to a v-dimensional output (i.e., class labels). The proposed monitoring technique can also be applied to ANNs, which solve regression

tasks by grouping the predicted values into a set of classes. In both cases, the dependent variables are class labels $y_t \in \{1, \ldots, v\}$, where the v-dimensional output of the algorithm contains the discriminant scores for each of the given classes. In supervised learning, we assume that true class labels are known for the training period $t = 1, \ldots, T$, which is used for estimating the parameters ϑ. Thus, $\hat{y}_t = \arg\max f(\boldsymbol{x}_t, \hat{\vartheta})$ for a set of input variables $\boldsymbol{x}_t$ is the prediction of ANNs, i.e., the most probable class, where $\hat{\vartheta}$ defines the learned parameters during the training, such that $\hat{y}_t$ coincides with y_t in most cases for all $t = 1, \ldots, T$.

There are several aspects to consider about the network's architecture when designing it, for instance, the number of hidden layers and neurons, the connections between the layers, and, fundamentally, which type of ANNs to use. To introduce these essential elements, a toy example of a feedforward ANN (FNN) which defines the basic family of ANN models is presented in Figure 1a. The network consists of four layers with two hidden layers, where each circle represents a neuron or node that stores a scalar value. Consequently, a layer is a k_i-dimensional vector with k_i being the number of nodes contained in the i-th layer. Here, the input layer is a k_1-dimensional vector $\boldsymbol{x}_t \in \mathbb{R}^7$ (first layer), and its first neuron is defined as $x_{t,1}$. Assuming we have a classification problem with two classes, we would construct an ANN with a one-dimensional output layer (last layer) that provides a score for the input to belong to class 2. When the output exceeds a threshold (often 0.5), the input is predicted as class 2, otherwise class 1. For a detailed description of the toy example set-up, see Section 4.3.

The changes in the number of nodes, layers, and types of connections lead to new ANN architectures. However, specialized types of ANN models were developed to handle high-dimensional data efficiently. For instance, convolutional ANNs enable image data processing, while recurrent ANNs are suitable for working with temporal sequences (cf. Shrestha and Mahmood, 2019). It is worth noting that designing ANNs usually follows the trial-and-error principle (Emambocus et al., 2023). Nevertheless, some strategies are recommended by the experts, e.g., how to start choosing hyperparameter values (cf. Smith, 2018). To obtain a broader overview of how to design and train ANNs, the reader can refer to Goodfellow et al. (2016).

Figure 1a illustrates how hidden layers in an ANN can reduce the dimensionality of the input sample. That produces valuable interim representations known as "embeddings" which we obtain before the activation function is applied, compressing the knowledge about the samples as shown in Figure 1b. They serve as a base for the change point detection procedure explained below.

(a) The FNN with two hidden layers in toy example

(b) Produced embeddings in toy example

Fig. 1: The FNN architecture displayed in **(a)** and embeddings from hidden layers (B) visualized in **(b)**. Gray-framed nodes represent the last hidden layer. Blue and green points denote training data references, while magenta points represent out-of-control data embeddings.

2.3 Change Point Detection

For monitoring ANN applications, we propose to use embeddings that usually have a vector form which we denote by $m_t \in \mathbb{R}^k$ with k being the dimension of the hidden layer that produces the embeddings. This representation is observed every time ANNs are applied, i.e., for historical (training) and new data. Thus, embeddings implicitly depend on x_t, but also on the fitted parameters $\hat{\vartheta}$, so that changes associated with ANN's data quality and performance can be detected.

In succeeding parts, we refer to the set $\{m_t : t = 1, \ldots, T\}$ as historical data with correctly known labels $\{y_t : t = 1, \ldots, T\}$ and to the set $\{m_i : i = T+1, T+2, \ldots\}$ as incoming data instance with the predicted class label $\hat{y}_i$ obtained from $f(m_i, \hat{\vartheta})$. Moreover, we consider that the true label of y_i is not available and historical data are stationary. Additionally, we assume that m_t follows a certain distribution $F_{m_t | y_t = c}$ depending on the true class y_t. These conditional distributions $F_{m_t | y_t = c}$ are denoted by Ξ_c for all classes $c \in \{1, \ldots, v\}$. Based on historical data, it is possible to estimate the distribution Ξ_c empirically that can be used to determine whether a possible change in the data stream occurred, i.e., whether the current observation m_i is generated from a different distribution than Ξ_{y_i}. Hence, we define a change point τ to arise if

$$
\boldsymbol{m}_i \quad \sim \quad
\begin{cases}
\Xi_{y_i} & \text{if } i < \tau \\
\Xi_v & \text{if } i \geq \tau.
\end{cases}
$$

In other words, a change point τ is the timestamp when the analysis of an embedding generated from the incoming data indicates that the data sample belongs to a different unknown distribution Ξ_v. Consequently, we can regard this situation as a change in the class definition and formulate a sequential hypothesis test of each sample i for detecting changes in the ANN application as follows

$$
\begin{aligned}
H_{0,i}: & \quad \Xi_{y_i} = \Xi_{\hat{y}_i} \quad\quad \text{against} \\
H_{1,i}: & \quad \text{there is a location shift and/or a scale increase in } \Xi_{y_i}.
\end{aligned}
$$

The alternative hypothesis could be true due to (1) misclassification by the model, i.e., $\hat{y}_i \neq y_i = a$, where $a \in \{1,\dots,v\}$, leading to $\Xi_{\hat{y}_i} \neq \Xi_a$, or (2) nonstationarity of the data stream, i.e., $\hat{y}_i \neq y_i = b$ with $b \notin \{1,\dots,v\}$ and $\Xi_{\hat{y}_i} \neq \Xi_b$ because $i \geq \tau$. The accurate distinction between those cases is crucial for reliable change point detection. Labeled data in Phase II helps achieve this, but a practical solution without labeled data is discussed in Section 4.4.3.

To test repeatedly whether the process is in control over time, i.e., whether the data assigned to a particular class does not deviate from the rest in this class, we can apply a multivariate control chart. Since the class distributions $\{\Xi_c : c = 1,\dots,v\}$ are unknown and can be estimated only empirically, we need a nonparametric monitoring technique that relaxes any distributional assumptions. Alternatively, the nonparametric kernel density estimates can be used, so-called Parzen window estimators (Parzen, 1962; Breiman et al., 1977). Moreover, kernel density estimates also have been proven beneficial for classification tasks (e.g., Ghosh et al., 2006).

Several multivariate nonparametric (distribution-free) charts are based on rank-based approaches. These approaches can be divided into two categories: control charts using longitudinal ranking and those employing cross-component ranking (Qiu, 2014). The first group includes componentwise longitudinal ranking (Boone and Chakraborti, 2012), spatial longitudinal ranking (Zou et al., 2012), and longitudinal ranking by data depth (Liu and Singh, 1993). While the first two subgroups are moment-dependent, control charts based on data depth offer flexibility by not imposing moment requirements. For instance, geometric and combinatorial depth functions, such as Simplicial depth and Halfspace depth, are considered and discussed in Section 3.1.

The depth-based control charts allow simultaneous monitoring for location shifts and scale increases in the process. The depth functions discussed in this work are affine invariant and satisfy im-

portant axioms such as monotonicity, convexity (except for Simplicial depth), and continuity (Mosler and Mozharovskyi, 2022). These characteristics make them suitable for our problem, leading us to employ nonparametric control charts based on data depth for online monitoring.

3 Monitoring Framework for ANN

As discussed in Section 2, ANNs input space is usually too complex to identify distributional changes directly from the input data (layer A, Figure 1a), whereas output does not always contain enough information for this purpose as we demonstrate in Section 4 (layer C, Figure 1a). A sufficient but not excessive amount of information can be obtained by intercepting the input propagation on an intermediate layer of the neural network (layer B, Figure 1a), e.g., to estimate prediction uncertainty which requires rarely accessible supervised training data (see Corbière et al., 2019). While several layers can be considered to account for more complex dependencies, one would normally take those which are closer to the output, achieving the highest dimensionality reduction (see, e.g., Parekh et al., 2021). Outputs of the intermediate layers constitute a Euclidean space, where—in the unsupervised setting—anomaly-detection techniques can be applied. These can constitute a neural network themselves (e.g., autoencoder), belong to (statistical) ML like a Local Outlier Factor (Breunig et al., 2000), one-class SVM (Schölkopf et al., 2001), isolation forest (Liu et al., 2008), or be based on data depth as proposed in this work.

The application of data depth in quality control was originally introduced by Liu and Singh (1993), resulting in the design of Shewhart-type multivariate nonparametric control charts based on the Simplicial depth (Liu, 1990, 1995). According to recent publications on data depth-based control charts (cf. Cascos and López-Díaz, 2018; Barale and Shirke, 2019; Pandolfo et al., 2021), the careful choice of data depth is crucial for satisfactory monitoring performance. Thus, we compare several notions of data depth and discuss their effectiveness in Section 4, looking at computational costs in Supplementary (Suppl.) Material, Part F.

3.1 Notion of Data Depth

A data depth is a concept for measuring the centrality of a multivariate observation m_i (cf. Zuo and Serfling, 2000; Liu et al., 2006; Mosler and Mozharovskyi, 2022) with respect to a given reference sample $R_c = \{m_t := 1, \ldots, |R|, y_t = c\}$, where $|R|$ defines its size which is the same for all considered

classes. In other words, it creates a center-outward ordering of points in the Euclidean space of any dimension. There are various notions of data depth, each of them providing a distinctive center-outward ordering of sample points in a multidimensional space. In this work, we consider four data depth notions: Halfspace, Mahalanobis, Projection, and Simplicial depths.

First, the Halfspace depth (originally known as "Tukey depth") introduced by Tukey (1975) and further developed by Donoho and Gasko (1992) is defined as the smallest number of data points in any closed halfspace with boundary hyperplane through $\boldsymbol{m}_i$ (Struyf and Rousseeuw, 1999). That is,

$$D_H^c(\boldsymbol{m}_i, R_c) = \frac{1}{|R|} \min_{\|\boldsymbol{p}\|=1} |\{b: \langle \boldsymbol{p}, \boldsymbol{m}_b \rangle \geq \langle \boldsymbol{p}, \boldsymbol{m}_i \rangle\}|,$$

where $|\cdot|$ denotes the cardinality of the set B with $\boldsymbol{m}_b \in R_c$, $\boldsymbol{p}$ are all possible directions with $\|\boldsymbol{p}\| = \sqrt{\langle p, p \rangle}$ being the Euclidean norm and $\langle \cdot, \cdot \rangle$ the inner product. In our work, we consider its robust version $\mathbf{HD}_r$ that is proposed by Ivanovs and Mozharovskyi (2021) and calculated approximately, offering some advantages in being strictly positive and continuous beyond the convex hull of the observed samples.

Second, we consider the Mahalanobis depth which is based on the Mahalanobis distance (cf. Mahalanobis, 1936). It is derived as

$$D_M^c(\boldsymbol{m}_i, R_c) = \frac{1}{1 + (\boldsymbol{m}_i - \boldsymbol{\mu_m})' \Sigma^{-1} (\boldsymbol{m}_i - \boldsymbol{\mu_m})},$$

where $\boldsymbol{\mu_m}$ is the mean vector of the embeddings in the reference sample and Σ^{-1} is the covariance matrix, estimated by the sample mean and the sample covariance matrix, respectively.

Third, the Projection depth proposed by Zuo and Serfling (2000) is specified as

$$D_P^c(\boldsymbol{m}_i, R_c) = \left(1 + \sup_{\|\boldsymbol{p}\|=1} \frac{|\langle \boldsymbol{p}, \boldsymbol{m}_i \rangle - \mathrm{med}(\langle \boldsymbol{p}, R_c \rangle)|}{\mathrm{MAD}(\langle \boldsymbol{p}, R_c \rangle)} \right)^{-1}$$

with $\langle \boldsymbol{p}, \boldsymbol{m}_i \rangle$ denoting the inner product and the projection of $\boldsymbol{m}_i$ to $\boldsymbol{p}$ if $\|\boldsymbol{p}\| = 1$. The notation $\mathrm{med}(E)$ defines the median of a univariate random variable E and $\mathrm{MAD}(E) = \mathrm{med}(|E - \mathrm{med}(E)|)$ is the median absolute deviation from the median. As the exact computation of the Projection depth is possible only at very high computational costs (cf. Mosler and Mozharovskyi, 2022), we use the algorithms that enable its calculation approximately. Dyckerhoff et al. (2021) provide the implementation and comparison of various algorithms. In our work, we study the performance of control charts based on three different algorithms to compute the Projection depth of both symmetric and asymmetric types. In particular, we consider coordinate descent ($\mathbf{PD}_1$), Nelder-Mead ($\mathbf{PD}_2$), and refined random search ($\mathbf{PD}_3$) for the symmetric type, and $\mathbf{PD}_1^a$, $\mathbf{PD}_2^a$, and $\mathbf{PD}_3^a$ for the asymmetric type.

Fourth, we calculate the Simplicial depth (Liu, 1990) as

$$D_S^c(\boldsymbol{m}_i, R_c) = \binom{|R|}{k+1}^{-1} \sum_{\diamond} I_{S(\boldsymbol{m}_t \mid t \in R_c,\, y_t = c)}(\boldsymbol{m}_i),$$

where $S(\boldsymbol{m}_t \mid t \in R_c, y_t = c)$ defines the open simplex consisting of vertices $\{\boldsymbol{m}_{t,1}, \ldots, \boldsymbol{m}_{t,k+1}\}$ from all observations t in the reference sample R_c. The $\diamond$ notation means that we validate all possible combinations to construct an open simplex with $(k+1)$ vertices. We specify $I_A(x)$ as the indicator function on a set A returning 1 if $x \in A$ and 0 otherwise. Both Simplicial (**SD**) and Mahalanobis (**MD**) depths are computed with algorithms provided by Pokotylo et al. (2019).

Related to the classification problem of multivariate data, there exist depth-based classifiers (cf. Vencálek, 2017) such as depth-vs-depth plot (DD-plot) designed by Li et al. (2012) or DD-alpha procedure proposed by Lange et al. (2014). Also, the field of outlier or anomaly detection is a widespread area for data depth usage (cf. Dang and Serfling, 2010; Baranowski et al., 2021). Combining these two perspectives, we apply a data depth-based Shewhart-type r control chart for single observations (see Sections 3.2 and 4) and a batch-wise Q control chart (see Suppl. Material, Part D) developed by Liu (1995) for detecting nonstationarity in a data stream.

3.2 The r Control Chart

A control chart is a graphical tool for monitoring processes by recording the performance of quality characteristics over time or sample number (Kan, 2003). The process is considered in control when the test statistic falls within the Upper and Lower Control Limits (*UCL* and *LCL*). Points outside this range indicate unusual variability, triggering a signal to investigate the out-of-control state of the process.

Usually, the application of control charts is divided into Phase I and Phase II. In Phase I, we collect the reference data, examine its quality and verify the process stability, then estimate model parameters if applicable and derive the values for the control limits (Jones-Farmer et al., 2014). The data in Phase I does not have to coincide with the full training data of the ANN but could rather be its subset. That is, the sets $R_c \subseteq \{\boldsymbol{m}_t : t = 1, \ldots, T,\ y_t = c\}$ of size $|R|$ create the Phase I data where it is essential to consider only correctly classified data samples. Successively, in Phase II, the control chart statistic is plotted for each embedding $\boldsymbol{m}_i$ with $i > T$. Figure 2 displays the introduced periods and sets.

Considering the r control chart proposed by Liu (1995), the scheme is based on ranks of multi-

Fig. 2: Summary of the introduced notation and data subdivision.

variate observations, which are obtained by computing data depth. To determine whether $\boldsymbol{m}_i$ belongs to Ξ_c, we use the following control chart statistic

$$r_{\cdot}^c(\boldsymbol{m}_i) = \frac{|\{D_{\cdot}^c(\boldsymbol{m}_t) \leq D_{\cdot}^c(\boldsymbol{m}_i) : t \in R_c,\ y_t = c\}|}{|\{t \in R_c :\ y_t = c\}|}$$

that defines the rank of the observed depth related to the observations in the reference sample with a class c. Thus, the r control chart monitors the values of $r_{\cdot}^c$ over time. Considering the interpretation of ranks, we can state that $r_{\cdot}^c(\boldsymbol{m}_i)$ reflects how outlying $\boldsymbol{m}_i$ is with respect to the reference sample. If $r_{\cdot}^c(\boldsymbol{m}_i)$ is high, then there is a considerable proportion of data in the reference sample that is more outlying compared to $\boldsymbol{m}_i$ (Liu, 1995).

Regarding the control limits, there is no need to introduce the *UCL* as $r_{\cdot}^c$ belongs to the continuous interval $[0,1]$. Considering the *LCL*, it coincides with the significance level of the hypothesis test, here defined as α. Thus, the process is considered to be out of control if $r_{\cdot}^c(\boldsymbol{m}_i) \leq \alpha$. The choice of α depends on the specification of Average Run Length (*ARL*) – the expected number of monitored data points required for the control chart to produce a signal (Stoumbos et al., 2001). In the case of the Shewhart-type control charts, the reciprocal of *ARL* corresponds to the false alarm rate (*FAR*) in the in-control state of a process. Technically speaking, since the r control chart is a Shewhart control

chart, $FAR = \alpha$, where α is interpreted as the probability of a false alarm in Phase I (Stoumbos et al., 2001).

According to Liu (1995), the r control chart can be applied with affine-invariant notions of data depth, explicitly mentioning Simplicial depth, Mahalanobis depth, and Halfspace depth. Since Projection depth is also affine-invariant (cf. Mosler, 2013), both r and Q control charts can be combined with each of the data depth functions introduced in Section 3.1.

4 Comparative Study

In the following section, we analyze the effectiveness of the proposed monitoring framework by designing a toy example and using real data. The discussion begins with a description of the considered benchmark methods, followed by the introduction of the selected performance measures in Section 4.2. Further, we compare the performances for toy example in Section 4.3 and for real data in Section 4.4. In addition, the construction of reference samples and the misclassification effect are examined in Section 4.4.3.

4.1 Benchmark Methods

Due to the independent development of comparable approaches (cf. Mozharovskyi, 2022; Yang et al., 2022b) and their focus on different scenarios/perspectives, there is no unified benchmark. Hence, to select a benchmark for ANN monitoring, we need to consider the available options. Three possibilities arise: inspecting the initial input, the model's embeddings, or the final output (softmax score in our case) represented by layers A, B, and C in Figure 1a. Monitoring the initial data quickly becomes limited due to the complexity of typical datasets analyzed by ANNs. On the contrary, using intermediate layers reduces data dimensionality and storage requirements since only the embeddings from the training phase need to be saved. Therefore, we only consider options B and C as monitoring options. Furthermore, as a benchmark, we focus on methods that can operate within the introduced framework. Specifically, we seek methods capable of detecting nonstationarity in (1) individual samples (batch size of one), (2) without the need for labels in Phase II, and (3) working with or without available time stamps. To ensure comparability, we utilize the same SPM framework of r control charts.

Based on the recent reviews (cf. Goldstein and Uchida, 2016; Villa-Pérez et al., 2021; Yang et al., 2022b), we choose a Kernel Density Estimation Outlier Score (**KDEOS**) defined by Schubert

et al. (2014), a distance-based Local Outlier Factor (**LOF**) developed by Breunig et al. (2000) and an ensemble-based outlier detection method such as isolation Forest (**iForest**) proposed by Liu et al. (2008) as a common benchmark coming from distribution- and distance-based methods. Both **LOF** and **KDEOS** compare the densities within local neighborhoods. However, while **LOF** is based on the reachability distance of the point to its neighborhood for density estimation, **KDEOS** uses classic kernel density estimates, e.g., based on Gaussian or Epanechnikov kernels. The **iForest** represents an ensemble of binary decision trees, where the points placed deeper in the trees are less likely to be outliers as they require more splits of space to isolate them. On the contrary, the samples which are allocated in shorter branches would rather be anomalous.

When considering option C, we compare our approach with monitoring the softmax scores. They are normalized between 0 and 1, and their length is equal to the number of neurons in the final layer. The neuron that has the maximum score corresponds to the predicted class. It is important to note that the softmax output is widely considered as a measure of the model's confidence (cf. Gawlikowski et al., 2021; Moon et al., 2020); however, there is substantial research into the area of alleviating overconfident prediction issue (Gawlikowski et al., 2021), e.g., by redesigning a loss function that leads to more trustful confidence estimates (Moon et al., 2020). Nevertheless, directly using the softmax output for nonstationarity detection is considered to perform reasonably well (Pearce et al., 2021). Thus, as benchmark techniques, we select Mahalanobis distance (**MDis**) which is well-known for detecting concept drift in similar settings (cf. Lee et al., 2018; Yang et al., 2022b), and a Natural Outlier Factor (**NOF**) based on Natural Neighbour principle, where calculation of the factor is parameterless (Huang et al., 2016).

4.2   Performance Measures

A well-operating control chart has a low false alarm rate (when it signals a change incorrectly) and a high rate of correctly detected out-of-control points. The performance of control charts is typically assessed by *ARL*. It measures the time until a false alarm when the process is in control and the speed at which the chart detects an actual change when the process is out of control. Alternatively, the False Alarm Rate (*FAR*) evaluates performance in Phase I, while the Signal Rate (*SR*) and Correct Detection Rate (*CDR*) assess performance in Phase II. All three metrics range between 0 and 1, providing the relative number of false or correct signals.

Regarding the *SR* value, we calculate a proportion of false alarms given the total length of the

Evaluation	Size $|R|$	Phase	Observed process	Metric	MD	SD	HD$_r$	PD$_1^a$	PD$_2^a$	PD$_3^a$	PD$_1$	PD$_2$	PD$_3$
Toy example $\boldsymbol{m}_i \in \mathbb{R}^3$	100	I	In-control	*FAR*	0.05	0.03	0.05	0.05	<u>0.05</u>	0.05	0.05	0.05	0.05
	100	II	In-control	*SR*	0.08	0.00	0.10	0.08	<u>0.04</u>	0.06	0.06	0.08	0.10
	100	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	<u>1.00</u>	1.00	0.10	0.06	0.14
Experiment 1 $\boldsymbol{m}_i \in \mathbb{R}^{16}$	2000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	<u>0.05</u>	0.05
	2000	II	In-control	*SR*	0.51	/	0.54	0.49	0.49	0.50	0.48	<u>0.47</u>	0.48
				SR|M	0.97	/	0.98	0.96	0.97	0.97	0.96	0.96	0.96
				SR|C	0.46	/	0.49	0.44	0.44	0.45	0.43	0.42	0.42
	2000	II	Out-of-control	*CDR*	0.92	/	0.93	0.92	0.92	0.92	0.91	<u>0.91</u>	0.89
	3000	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	<u>0.05</u>	0.05
	3000	II	In-control	*SR*	0.43	/	0.46	0.40	0.41	0.41	0.41	<u>0.39</u>	0.39
				SR|M	0.95	/	0.95	0.92	0.92	0.93	0.93	0.92	0.92
				SR|C	0.38	/	0.40	0.35	0.35	0.35	0.35	0.33	0.33
	3000	II	Out-of-control	*CDR*	0.88	/	0.90	0.86	0.87	0.86	0.87	<u>0.86</u>	0.84
	4000	I	In-control	*FAR*	0.05	/	0.05	0.05	<u>0.05</u>	0.05	0.05	0.05	0.05
	4000	II	In-control	*SR*	0.34	/	0.37	0.32	<u>0.31</u>	0.31	0.33	0.31	0.30
				SR|M	0.88	/	0.91	0.83	0.84	0.83	0.86	0.83	0.83
				SR|C	0.29	/	0.32	0.26	0.26	0.26	0.27	0.25	0.25
	4000	II	Out-of-control	*CDR*	0.79	/	0.83	0.78	<u>0.79</u>	0.78	0.78	0.78	0.73

Table 1: Performance of r control charts ($\alpha = 0.05$) in toy example and in Experiment 1 with reference samples R being predicted classes. The underlined numbers indicate the suggested method for an entire monitoring period, based on the trade-off between *SR* and *CDR*. We compute **SD** for $\mathbb{R}^3$ only, as computational complexity is $O(n^{k+1})$.

considered in-control part. In the case of the *CDR* value, it is computed as a proportion of correctly detected out-of-control data points given the total length of the designed out-of-control part. To account for a possible discrepancy between the class proportions of the predicted data in Phase II, we calculate the weighted mean of the occurred signals, accounting for the number of data points in each predicted class within the observed period. In the case of *FAR*, we simply use the sample mean because the class or reference sample sizes are identical.

If a tested control chart operates as desired, then *FAR* of Phase I equals the chosen probability of a false alarm α. In Phase II, ideally, we would expect *SR* to be similar to *FAR* for the in-control samples (neglecting the potential misclassification effect), while the *CDR* should be as large as possible for the out-of-control samples. If the *CDR* is low, i.e., close to 0, we conclude that the control chart does not accomplish its primary purpose – to detect nonstationarity in a data stream.

4.3 Toy Example

To present our idea in a controllable environment, we create a toy example using the ANN architecture presented in Figure 1a. We simulate two 7-dimensional Gaussian random variables $\boldsymbol{x}_t^{(1)}$ and $\boldsymbol{x}_t^{(2)}$ with $\boldsymbol{\mu}_1 = \boldsymbol{0}$ and $\boldsymbol{\mu}_2 = 10 \cdot \boldsymbol{1}_7$, where $t = 1, \ldots, 100$ and $\boldsymbol{1}_n$ is the n-dimensional vector of ones. Both

Evaluation	Size $|R|$	Phase	Observed process	Metric	$m_i \in \mathbb{R}$		$m_i \in \mathbb{R}^{16}$			
					MDis	NOF	KDEOS	LOF	iForest	$\mathbf{PD}_2^{(a)}$
	100	I	In-control	*FAR*	0.05	<u>0.05</u>	0.05	<u>0.05</u>	0.05	<u>0.05</u>
Toy example	100	II	In-control	*SR*	0.08	<u>0.04</u>	0.00	<u>0.04</u>	0.12	<u>0.04</u>
	100	II	Out-of-control	*CDR*	1.00	<u>1.00</u>	1.00	<u>1.00</u>	1.00	<u>1.00</u>
	2000	I	In-control	*FAR*	0.05	0.05	0.05	0.05	0.05	<u>0.05</u>
	2000	II	In-control	*SR*	0.57	0.60	0.07	0.56	0.53	<u>0.47</u>
	2000	II	Out-of-control	*CDR*	0.92	0.92	0.05	0.93	0.93	<u>0.91</u>
	3000	I	In-control	*FAR*	0.05	0.05	0.05	0.05	0.05	<u>0.05</u>
Experiment 1	3000	II	In-control	*SR*	0.44	0.47	0.07	0.48	0.46	<u>0.39</u>
	3000	II	Out-of-control	*CDR*	0.87	0.86	0.07	0.87	0.89	<u>0.86</u>
	4000	I	In-control	*FAR*	0.05	0.05	0.05	0.05	0.05	<u>0.05</u>
	4000	II	In-control	*SR*	0.33	0.36	0.07	0.39	0.38	<u>0.31</u>
	4000	II	Out-of-control	*CDR*	0.79	0.78	0.10	0.77	0.83	<u>0.79</u>

Table 2: Comparison study: Performance of *r* control charts ($\alpha = 0.05$) in toy example and in Experiment 1. In the case of **MDis** and **NOF**, the data points represent the model's softmax output ($m_i \in \mathbb{R}$). The underlined numbers indicate the suggested method based on the trade-off between *SR* and *CDR*.

variance-covariance matrices Σ_1 and Σ_2 have $\sigma_{ii} = 1$ but with $\sigma_{i-1,j} = \sigma_{i,j-1} = 0.3$ in the case of Σ_1, and $\sigma_{i-1,j} = \sigma_{i,j-1} = -0.3$ in case of Σ_2 for all $i, j = 1, \ldots, 7$ (in respective cases $i, j > 1$), where the remaining entries are zero. Considering the out-of-control data, we sample from a new multivariate Gaussian distribution with $\mu_\tau = 5 \cdot 1_7$ and Σ_1.

We use a reference sample of size $|R| = 100$ for each of the classes, i.e., the entire training data because there were no misclassified data points. In Phase II, the in-control data corresponds to 50 new observations with the same distribution as used for training, while the out-of-control 50 data points correspond to the out-of-control distribution. The embedding layer consists of three neurons, i.e. $m_i \in \mathbb{R}^3$. The visualization of the embeddings that correspond to both reference samples and out-of-control data is displayed in Figure 1b.

Table 1 summarizes the results from depth-based control charts. As we can see, all versions apart from symmetric Projection and Simplicial depths can be successfully applied in this setting. The reason why symmetric Projection depths fail is the asymmetric distribution of the processed data (see Figure 1b). Regarding the Simplicial depth, there are 24% of data points in the reference sample of class 2 with $\mathbf{SD}(m_t) = 0$. That can be explained by Simplicial depth assigning zero to every point in the space outside the sample's convex hull (Francisci et al., 2019). Moreover, because all out-of-control samples received the predictions of class 2, the **SD** could not detect the out-of-control samples. Comparing the best result from Table 1 which is $\mathbf{PD}_2^a$ to the benchmark in Table 2, we notice

Experiment	1	2	3
Complexity	High	Medium	Low
Data type	Image	Sentence	Signal in $[0, 1]$
Type of NN	CNN	LSTM	FNN
Number of classes	10	4	2
Phase I, in-control (Training data)	50000	2800	170
Phase II, in-control (Test data)	10000	600	38
Phase II, out-of-control (Nonstationary data)	400	60	30
Results	Section 4	Suppl. Material, Part B	Suppl. Material, Part C

Table 3: Summary of experiments: Data size indicates the total number of samples across all classes. Balanced datasets are used for training the ANN, ensuring an equal representation of samples from each class.

that the **LOF** and **NOF** achieved similar results, while the **KDEOS** and **iForest** did not hold their size of $\alpha = 0.05$ in Phase II.

4.4 Experiments with Real Data

We conduct in total three experiments. In decreasing complexity, the first experiment is about a ten-class classification of images, applying Convolutional ANN (CNN), followed by a four-class classification of questions, using ANN with a Long Short-Term Memory layer (LSTM) and finished with a binary classification of sonar data performed with an FNN. Table 3 provides a summary of the conducted experiments. The models' training aims to maximize overall classification accuracy, respectively tuning the hyperparameters and the ANN architectures. For the sake of brevity, we only report the results of Experiment 1 below and include the results of the other two experiments in Suppl. Material.

4.4.1 Multiclass Classification of Images

In this experiment, we work with the CIFAR-10 dataset[1] containing color images of the size 32×32 pixels (Krizhevsky et al., 2009), which is often applied for testing new out-of-distribution detection methods (cf. Yang et al., 2022a). In total, there are 60000 images which correspond to 6000 pictures

---

[1] https://www.cs.toronto.edu/~kriz/cifar.html

Fig. 3: Image examples and class labels of the CIFAR-10 dataset.

per class. Figure 3 shows examples of each category. For the out-of-control samples, we consider the CIFAR-100 dataset[1] that has 100 image groups, selecting four distinctive classes, namely "Kangaroo", "Butterfly", "Train" and "Rocket". From each category we randomly chose 100 images, having in total 400 samples for the out-of-control part in Phase II.

To construct a classifier for predicting to which of the ten groups an input image belongs, we train a CNN with a deep layer aggregation structure as proposed by Yu et al. (2018). A specification of such architecture is a tree-structured hierarchy of operations to aggregate the extracted features from different model stages. For a detailed introduction to CNNs, we direct to O'Shea and Nash (2015). The embedding layer has 16 neurons, so we obtain a monitoring task of $m_i \in \mathbb{R}^{16}$. Regarding the training results after 88 epochs, the achieved accuracy on the test images was 90.43%.

4.4.2 Choice of Reference Samples and Monitoring Results

Below, we investigate the effect of different reference samples, particularly concentrating on their size. To guarantee a well-chosen reference sample for each class, we construct it by choosing $|R|$ data points that obtained the highest softmax scores. The analysis of applying randomly created reference samples is provided in Suppl. Material, Part A. To illustrate the application of the r control chart in Figure 4, we depict $\mathbf{PD}_2$ with $|R| = 4000$. While we have 3 signals in Phase I (green points), where $FAR = 5\%$, a considerably larger number of signals is observed in Phase II without novelty (i.e., in-control, purple points). A substantial part of these signals occurred on the misclassified samples, marked by the asterisks. In the out-of-control part in Phase II (red points), we notice the highest number of signals, signifying correctly detected nonstationarity.

18

Fig. 4: An example of the *r* control chart based on $\mathbf{PD}_2$ using the data from Experiment 1. Colors correspond to the phases illustrated in Figure 2. The dashed line represents the control limit $\alpha = 0.05$, the signals are shown in dark red, and the misclassified samples in Phase II (in-control) are indicated with an asterisk.

Looking at the results shown in Table 2, we can recognize the following patterns: First, with increasing reference sample size, the number of false alarms in Phase II decreases. For instance, **MD** and **HD**$_r$ lead to *SR* being over 50% when $|R| = 2000$, but we observe improvements by over 15% when the size of the reference sample increases. Second, the larger the reference sample, the less precise becomes the out-of-control detection. Here, the *CDR* values remain moderately high while doubling the size of the reference samples. Hence, it is beneficial to agree on such a reference sample size that slightly decreases *CDR* and, at the same time, improves the performance of the control chart during the in-control state. With this strategy, both $\mathbf{PD}_2^a$ and $\mathbf{PD}_2$ suit the entire monitoring period well.

Similar behavior can be noticed for the benchmark methods presented in Table 2. Comparing the best depth-based monitoring result of $\mathbf{PD}_2^{(a)}$ with the benchmark, we note that it outperforms all other algorithms. Nevertheless, the *SR* values remain generally high, which could be due to misclassified observations being flagged as anomalous samples or because the dispersion of the test data is large compared to the reference samples. To better understand these issues, we analyze the data in Phase II more closely in the subsequent part.

19

4.4.3 Misclassification and Data Diagnostic

To investigate the effect of misclassification, we refer to the wrongly classified images from the test data (Phase II, in-control), which constitute 958 data points. By calculating the signal rates conditional on misclassified $SR|M$ and correctly classified samples $SR|C$ for each depth notion and reference sample size in Table 2, we find out that the average $SR|M$ is 91.58%. At the same time, $SR|C$ values are considerably lower and approach 25% for bigger reference samples, compared to the original SR results.

To investigate another reason for high SR values, we visualize the in-control data in Figure 5. We use Radial Coordinate visualization (Radviz), where the variables are referred to as anchors, being evenly distributed around a unit circle (cf. Hoffman et al., 1999; Caro et al., 2010; Abraham et al., 2017). Their order is optimized to place highly correlated variables next to each other. Correspondingly, data points are projected to positions close to the variables that have a higher influence on them. As we can see in Figure 5, there is a comparably large section opposite the anchor V_{15} where the test data does not overlap with any of the reference samples. At the same time, a fraction of the out-of-control samples is located within reference samples, leading to more challenging detection. Hence, this analysis and the evaluation of the misclassification effect could facilitate the understanding of $SR > FAR$ and provide insight into CDR.

A possible solution for mitigating the misclassification effect and challenges in choosing a representative reference sample for each class could be a creation of a merged reference sample. In other words, we could neglect the condition of having class-specified reference samples and perform the computation of depths with respect to a grouped reference sample only. However, according to the results presented in Suppl. Material, Part E, such construction of reference samples could eliminate misclassification and sample selection issues but work only for less complex cases. Moreover, the computation time would increase rapidly for high-dimensional problems (see Suppl. Material, Part F), meaning that the proposed framework of using individual reference samples for each class would also be more suitable from this perspective.

4.5 Practical Recommendations

Overall, we notice that the asymmetric Projection depth works most reliably among the examined depths. The reason for that is twofold: First, it considers the geometry of the points, i.e., their asym-

Fig. 5: Visualization of the data from Experiment 1 with $|R| = 2000$. $V_1, \ldots, V_{16}$ define anchors which correspond to neurons that produced embeddings $\boldsymbol{m}_i \in \mathbb{R}^{16}$. Density contour plots outline respective classes.

metric positioning. Second, as we usually aim to diagnose the outlyingness of the points that are outside of the convex hull, we need to be able to order them. By obtaining positive depth values outside the convex hull, we can better recognize which points are anomalous. On the contrary, the Simplicial or symmetric Projection depths underperform, if many points are placed outside the convex hull (an issue for **SD**) or the data is asymmetrically spread (an issue for **PD**).

As soon as a change has been detected, various actions could be implemented. Lu et al. (2018) introduce the idea of "Concept Drift Understanding" before its adaptation. They stress that it is vital to answer how severe and in which data region the concept drift occurred before implementing further actions. Afterward, the model can be either adjusted or rebuilt, resulting in a new cycle of training and validation.

5 Conclusion

The models based on ANNs have contributed to recent advances in various disciplines. However, the impressive results that were achieved with their deployment mask the necessity of the model's control and monitoring, meaning that critical flaws could occur without any notice by the expert.

This work proposes a monitoring procedure designed for ANN applications that applies a non-parametric multivariate control chart based on ranks and data depths. The core idea is to monitor the

low-dimensional representation of input data called embeddings that are generated by ANNs. The proposed monitoring methodology has great potential and often outperforms benchmark methods in realistic experiments. Comparing different data depth notions under the trade-off between computation time and monitoring effectiveness, we recommend the asymmetric Projection depth using the Nelder-Mead algorithm. In case the embeddings are scattered symmetrically, symmetric Projection depth can be used instead.

Furthermore, we investigated the influence of the reference sample size and the method of how it was constructed. It could be shown that a bigger reference sample is not automatically related to a better detection capability. In particular, there are open questions about attaining a reliable reference sample. It is advisable to research in more detail how the SPM techniques, such as the multivariate mean-rank chart (Bell et al., 2014), could support the analysis of Phase I data. Also, it is subject to future research when the reference sample should be updated or continually augmented with recent observations while preventing contamination with the out-of-control data.

In our experiments, the data points of the training and the test datasets are chosen following a convention by randomly dividing the dataset. In the future, we recommend examining whether an optimal splitting of data into training and testing, which preserves distributional similarity, improves the performance of models based on AI as well as leads to more reliable monitoring (cf. Vakayil and Joseph, 2022). Additionally, the field of data splitting and data compression, i.e., how to find a trade-off between reliable but fast training and a well-chosen training set that uses only a fraction of the initial dataset, is relevant for future research.

We have also shown that our approach would achieve a better performance if additional misclassification information were available. In general, understanding when a data point has obtained a wrong prediction is indispensable and requires an additional method to be developed that could be later combined with our monitoring procedure. Moreover, one could investigate whether subdividing the data stream in moving windows could enhance the monitoring performance.

In the empirical study, we consider well-balanced classification problems. However, class imbalance is a frequent challenge in training ANNs and needs to be considered in future research (cf. Ghazikhani et al., 2013). Whether the proposed methodology could be applied to monitoring semi-supervised or unsupervised learning models remains also open. Despite of challenges in designing a universal monitoring scheme for AI-based approaches, extending the presented framework to other AI algorithms seems to be a promising field. Additionally, there are different types of concept drift or

nonstationarity (cf. Hu et al., 2020), meaning that further development of methods and their comparison is of practical importance.

In summary, choosing a particular control chart depends on the specific problem, requiring a compromise between the number of signals in Phase II (In-control) and correctly detected out-of-control samples. As soon as one concludes what is important, namely computation time, robustness, or low variance, the proposed monitoring approach can be customized to satisfactorily support applications involving ANN.

References

Abraham, Y., Gerrits, B., Ludwig, M.-G., Rebhan, M., and Gubser Keller, C. (2017). Exploring Glucocorticoid Receptor Agonists Mechanism of Action Through Mass Cytometry and Radial Visualizations. Cytometry Part B: Clinical Cytometry, 92(1):42–56.

Afshani, P., Sheehy, D. R., and Stein, Y. (2016). Approximating the simplicial depth in high dimensions. In The European Workshop on Computational Geometry.

Aldridge, I. and Avellaneda, M. (2019). Neural Networks in Finance: Design and Performance. The Journal of Financial Data Science, 1(4):39–62.

Ali, S., Pievatolo, A., and Göb, R. (2016). An Overview of Control Charts for High-quality Processes. Quality and reliability engineering international, 32(7):2171–2189.

Angermueller, C., Pärnamaa, T., Parts, L., and Stegle, O. (2016). Deep learning for computational biology. Molecular systems biology, 12(7):878.

Apsemidis, A., Psarakis, S., and Moguerza, J. M. (2020). A review of machine learning kernel methods in statistical process monitoring. Computers & Industrial Engineering, 142:106376.

Baena-Garcıa, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R., and Morales-Bueno, R. (2006). Early Drift Detection Method. In Fourth international workshop on knowledge discovery from data streams, volume 6, pages 77–86.

Ball, N. M., Brunner, R. J., Myers, A. D., and Tcheng, D. (2006). Robust machine learning applied to astronomical data sets. I. star-galaxy classification of the Sloan Digital Sky Survey DR3 using decision trees. The Astrophysical Journal, 650(1):497.

Barale, M. and Shirke, D. (2019). Nonparametric control charts based on data depth for location parameter. Journal of Statistical Theory and Practice, 13(3):1–19.

Baranowski, J., Dudek, A., and Mularczyk, R. (2021). Transient Anomaly Detection Using Gaussian Process Depth Analysis. In 2021 25th International Conference on Methods and Models in Automation and Robotics (MMAR), pages 221–226. IEEE.

Bell, R. C., Jones-Farmer, L. A., and Billor, N. (2014). A Distribution-Free Multivariate Phase I Location Control Chart for Subgrouped Data from Elliptical Distributions. Technometrics, 56(4):528–538.

Bifet, A. and Gavalda, R. (2007). Learning from time-changing data with adaptive windowing. In Proceedings of the 2007 SIAM international conference on data mining, pages 443–448.

Bifet, A., Gavalda, R., Holmes, G., and Pfahringer, B. (2018). Machine learning for data streams: with practical examples in MOA. MIT press.

Boone, J. and Chakraborti, S. (2012). Two simple Shewhart-type multivariate nonparametric control charts. Applied Stochastic Models in Business and Industry, 28(2):130–140.

Breiman, L., Meisel, W., and Purcell, E. (1977). Variable kernel estimates of multivariate densities. Technometrics, 19(2):135–144.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). LOF: identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD international conference on Management of data, pages 93–104.

Caro, L. D., Frias-Martinez, V., and Frias-Martinez, E. (2010). Analyzing the role of dimension arrangement for data visualization in radviz. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 125–132. Springer.

Cascos, I. and López-Díaz, M. (2018). Control charts based on parameter depths. Applied Mathematical Modelling, 53:487–509.

Celano, G., Castagliola, P., Fichera, S., and Nenes, G. (2013). Performance of t control charts in short runs with unknown shift sizes. Computers & Industrial Engineering, 64(1):56–68.

Chiroma, H., Abdullahi, U. A., Alarood, A. A., Gabralla, L. A., Rana, N., Shuib, L., Hashem, I. A. T., Gbenga, D. E., Abubakar, A. I., Zeki, A. M., et al. (2018). Progress on Artificial Neural Networks for Big Data Analytics: A Survey. IEEE Access, 7:70535–70551.

Cook, R. D. and Ni, L. (2005). Sufficient dimension reduction via inverse regression: A minimum discrepancy approach. Journal of the American Statistical Association, 100(470):410–428.

Corbière, C., Thome, N., Bar-Hen, A., Cord, M., and Pérez, P. (2019). Addressing Failure Prediction by Learning Model Confidence. Advances in Neural Information Processing Systems, 32.

Dang, X. and Serfling, R. (2010). Nonparametric depth-based multivariate outlier identifiers, and masking robustness properties. Journal of Statistical Planning and Inference, 140(1):198–213.

Demšar, J. and Bosnić, Z. (2018). Detecting concept drift in data streams using model explanation. Expert Systems with Applications, 92:546–559.

Donoho, D. L. and Gasko, M. (1992). Breakdown properties of location estimates based on halfspace depth and projected outlyingness. The Annals of Statistics, pages 1803–1827.

Dua, D. and Graff, C. (2019). UCI machine learning repository.

Dyckerhoff, R., Mozharovskyi, P., and Nagy, S. (2021). Approximate computation of projection depths. Computational Statistics & Data Analysis, 157:107166.

Emambocus, B. A. S., Jasser, M. B., and Amphawan, A. (2023). A survey on the optimization of artificial neural networks using swarm intelligence algorithms. IEEE Access, 11:1280–1294.

Fang, Z., Li, Y., Lu, J., Dong, J., Han, B., and Liu, F. (2022). Is out-of-distribution detection learnable? arXiv preprint arXiv:2210.14707.

Forrest, S., Perelson, A. S., Allen, L., and Cherukuri, R. (1994). Self-nonself discrimination in a computer. In Proceedings of 1994 IEEE computer society symposium on research in security and privacy, pages 202–212. IEEE.

Fort, S., Ren, J., and Lakshminarayanan, B. (2021). Exploring the limits of out-of-distribution detection. Advances in Neural Information Processing Systems, 34:7068–7081.

Francisci, G., Nieto-Reyes, A., and Agostinelli, C. (2019). Generalization of the simplicial depth: no vanishment outside the convex hull of the distribution support. arXiv preprint arXiv:1909.02739.

Gama, J., Medas, P., Castillo, G., and Rodrigues, P. (2004). Learning with drift detection. In Brazilian symposium on artificial intelligence, pages 286–295. Springer.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. (2014). A survey on concept drift adaptation. ACM computing surveys (CSUR), 46(4):1–37.

Gao, X., Pishdad-Bozorgi, P., Shelden, D. R., and Hu, Y. (2019). Machine learning applications in facility life-cycle cost analysis: A review. Computing in Civil Engineering 2019: Smart Cities, Sustainability, and Resilience, pages 267–274.

Garcia, K. D., Poel, M., Kok, J. N., and de Carvalho, A. C. (2019). Online clustering for novelty detection and concept drift in data streams. In EPIA Conference on Artificial Intelligence, pages 448–459. Springer.

Gawlikowski, J., Tassi, C. R. N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A., Triebel, R., Jung, P., Roscher, R., et al. (2021). A survey of uncertainty in deep neural networks. arXiv preprint arXiv:2107.03342.

Gemaque, R. N., Costa, A. F. J., Giusti, R., and Dos Santos, E. M. (2020). An overview of unsupervised drift detection methods. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 10(6):e1381.

Ghazikhani, A., Monsefi, R., and Yazdi, H. S. (2013). Ensemble of online neural networks for non-stationary and imbalanced data streams. Neurocomputing, 122:535–544.

Ghosh, A. K., Chaudhuri, P., and Sengupta, D. (2006). Classification using kernel density estimates: Multiscale analysis and visualization. Technometrics, 48(1):120–132.

Goldberg, Y. (2016). A primer on neural network models for natural language processing. Journal of Artificial Intelligence Research, 57:345–420.

Goldstein, M. and Uchida, S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS one, 11(4):e0152173.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep learning. MIT press.

Gorman, R. P. and Sejnowski, T. J. (1988). Analysis of hidden units in a layered network trained to classify sonar targets. Neural networks, 1(1):75–89.

Guttormsson, S. E., Marks, R., El-Sharkawi, M., and Kerszenbaum, I. (1999). Elliptical novelty grouping for on-line short-turn detection of excited running rotors. IEEE Transactions on Energy Conversion, 14(1):16–22.

Haque, A., Khan, L., Baron, M., Thuraisingham, B., and Aggarwal, C. (2016). Efficient handling of concept drift and concept evolution over stream data. In 2016 IEEE 32nd International Conference on Data Engineering (ICDE), pages 481–492. IEEE.

Heipke, C. and Rottensteiner, F. (2020). Deep learning for geometric and semantic tasks in photogrammetry and remote sensing. Geo-spatial Information Science, 23(1):10–19.

Hendrycks, D. and Gimpel, K. (2016). A baseline for detecting misclassified and out-of-distribution examples in neural networks. arXiv preprint arXiv:1610.02136.

Hermans, M. and Schrauwen, B. (2013). Training and analysing deep recurrent neural networks. Advances in neural information processing systems, 26:190–198.

Hoffman, P., Grinstein, G., and Pinkney, D. (1999). Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations. In Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management, pages 9–16.

Hu, H., Kantardzic, M., and Sethi, T. S. (2020). No free lunch theorem for concept drift detection in streaming data classification: A review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 10(2):e1327.

Huang, J., Zhu, Q., Yang, L., and Feng, J. (2016). A non-parameter outlier detection algorithm based on natural neighbor. Knowledge-Based Systems, 92:71–77.

Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I., and Tygar, J. D. (2011). Adversarial machine learning. In Proceedings of the 4th ACM workshop on Security and artificial intelligence, pages 43–58.

Ivanovs, J. and Mozharovskyi, P. (2021). Distributionally robust halfspace depth. arXiv preprint arXiv:2101.00726.

Jones-Farmer, L. A., Woodall, W. H., Steiner, S. H., and Champ, C. W. (2014). An Overview of Phase I Analysis for Process Improvement and Monitoring. Journal of Quality Technology, 46(3):265–280.

Kan, S. H. (2003). Metrics and models in software quality engineering. Addison-Wesley Professional.

Kim, Y. and Park, C. H. (2017). An efficient concept drift detection method for streaming data under limited labeling. IEICE Transactions on Information and systems, 100(10):2537–2546.

Klinkenberg, R. and Joachims, T. (2000). Detecting concept drift with support vector machines. In ICML, pages 487–494.

Klinkenberg, R. and Renz, I. (1998). Adaptive information filtering: Learning in the presence of concept drifts. Learning for Text Categorization, pages 33–40.

Krawczyk, B. and Woźniak, M. (2015). One-class classifiers with incremental learning and forgetting for data streams with concept drift. Soft Computing, 19(12):3387–3400.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images. Toronto, ON, Canada.

Kuncheva, L. I. (2009). Using control charts for detecting concept change in streaming data. Bangor University, page 48.

Lange, T., Mosler, K., and Mozharovskyi, P. (2014). Fast nonparametric classification based on data depth. Statistical Papers, 55(1):49–69.

Lee, K., Lee, K., Lee, H., and Shin, J. (2018). A simple unified framework for detecting out-of-distribution samples and adversarial attacks. Advances in neural information processing systems, 31.

Lee, S., Kwak, M., Tsui, K.-L., and Kim, S. B. (2019). Process monitoring using variational autoencoder for high-dimensional nonlinear processes. Engineering Applications of Artificial Intelligence, 83:13–27.

Li, J., Cuesta-Albertos, J. A., and Liu, R. Y. (2012). DD-classifier: Nonparametric classification procedure based on DD-plot. Journal of the American statistical association, 107(498):737–753.

Li, K.-C. (1991). Sliced inverse regression for dimension reduction. Journal of the American Statistical Association, 86(414):316–327.

Li, L. (2007). Sparse sufficient dimension reduction. Biometrika, 94(3):603–613.

Li, L. and Nachtsheim, C. J. (2006). Sparse sliced inverse regression. Technometrics, 48(4):503–510.

Li, P., Wu, X., Hu, X., and Wang, H. (2015). Learning concept-drifting data streams with random ensemble decision trees. Neurocomputing, 166:68–83.

Lin, Q., Zhao, Z., and Liu, J. S. (2019). Sparse sliced inverse regression via lasso. Journal of the American Statistical Association, 114(528):1726–1739.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining, pages 413–422.

Liu, R. Y. (1990). On a notion of data depth based on random simplices. The Annals of Statistics, pages 405–414.

Liu, R. Y. (1995). Control charts for multivariate processes. Journal of the American Statistical Association, 90(432):1380–1387.

Liu, R. Y., Serfling, R. J., and Souvaine, D. L. (2006). Data depth: robust multivariate analysis, computational geometry, and applications, volume 72. American Mathematical Soc.

Liu, R. Y. and Singh, K. (1993). A quality index based on data depth and multivariate rank tests. Journal of the American Statistical Association, 88(421):252–260.

Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., and Zhang, G. (2018). Learning under concept drift: A review. IEEE transactions on knowledge and data engineering, 31(12):2346–2363.

Mahalanobis, P. C. (1936). On the Generalised Distance in Statistics. In Proceedings of the National Institute of Sciences of India, volume 2, pages 49–55.

Markou, M. and Singh, S. (2003a). Novelty detection: a review—part 1: statistical approaches. Signal Processing, 83(12):2481–2497.

Markou, M. and Singh, S. (2003b). Novelty detection: a review—part 2:: neural network based approaches. Signal Processing, 83(12):2499–2521.

Masud, M. M., Gao, J., Khan, L., Han, J., and Thuraisingham, B. (2009). Integrating novel class detection with classification for concept-drifting data streams. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2009, Bled, Slovenia, September 7-11, 2009, Proceedings, Part II 20, pages 79–94. Springer.

Mejri, D., Limam, M., and Weihs, C. (2017). Combination of Several Control Charts Based on Dynamic Ensemble Methods. Mathematics and Statistics, 5(3):117–129.

Mejri, D., Limam, M., and Weihs, C. (2021). A new time adjusting control limits chart for concept drift detection. IFAC Journal of Systems and Control, 17:100170.

Montgomery, D. C. (2020). Introduction to statistical quality control. John Wiley & Sons.

Moon, J., Kim, J., Shin, Y., and Hwang, S. (2020). Confidence-aware learning for deep neural networks. In Proceedings of the 37th International Conference on Machine Learning, volume 119, pages 7034–7044. PMLR.

Mosler, K. (2013). Depth statistics. Robustness and complex data structures: Festschrift in Honour of Ursula Gather, pages 17–34.

Mosler, K. and Mozharovskyi, P. (2022). Choosing among notions of multivariate depth statistics. Statistical Science, 37(3):348–368.

Mozharovskyi, P. (2022). Anomaly detection using data depth: multivariate case. arXiv preprint arXiv:2210.02851.

Nammous, M. K. and Saeed, K. (2019). Natural language processing: speaker, language, and gender identification with LSTM. In Advanced Computing and Systems for Security, pages 143–156. Springer.

Nishida, K. and Yamauchi, K. (2007). Detecting concept drift using statistical testing. In Discovery science, volume 4755, pages 264–269. Springer.

O'Shea, K. and Nash, R. (2015). An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458.

Otter, D. W., Medina, J. R., and Kalita, J. K. (2020). A survey of the usages of deep learning for natural language processing. IEEE Transactions on Neural Networks and Learning Systems, 32(2):604–624.

Pandolfo, G., Iorio, C., Staiano, M., Aria, M., and Siciliano, R. (2021). Multivariate process control charts based on the L^p depth. Applied Stochastic Models in Business and Industry, 37(2):229–250.

Parekh, J., Mozharovskyi, P., and d'Alché-Buc, F. (2021). A framework to learn with interpretation. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, Advances in Neural Information Processing Systems, volume 34, pages 24273–24285. Curran Associates, Inc.

Parzen, E. (1962). On estimation of a probability density function and mode. The annals of mathematical statistics, 33(3):1065–1076.

Pearce, T., Brintrup, A., and Zhu, J. (2021). Understanding softmax confidence and uncertainty. arXiv preprint arXiv:2106.04972.

Perdikis, T. and Psarakis, S. (2019). A survey on multivariate adaptive control charts: Recent developments and extensions. Quality and Reliability Engineering International, 35(5):1342–1362.

Piano, L., Garcea, F., Gatteschi, V., Lamberti, F., and Morra, L. (2022). Detecting Drift in Deep Learning: A Methodology Primer. IT Professional, 24(5):53–60.

Pidhorskyi, S., Almohsen, R., and Doretto, G. (2018). Generative probabilistic novelty detection with adversarial autoencoders. Advances in Neural Information Processing Systems, 31.

Pokotylo, O., Mozharovskyi, P., and Dyckerhoff, R. (2019). Depth and Depth-Based classification with R Package ddalpha. Journal of Statistical Software, 91(5):1–46.

Psarakis, S. (2011). The use of neural networks in statistical process control charts. Quality and Reliability Engineering International, 27(5):641–650.

Psarakis, S. (2015). Adaptive control charts: Recent developments and extensions. Quality and Reliability Engineering International, 31(7):1265–1280.

Qiu, P. (2014). Introduction to statistical process control. CRC press.

Roberts, S. and Tarassenko, L. (1994). A probabilistic resource allocating network for novelty detection. Neural Computation, 6(2):270–284.

Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., and Williamson, R. (2001). Estimating the support of a high-dimensional distribution. Neural Computation, 13(7):1443–1471.

Schubert, E., Zimek, A., and Kriegel, H.-P. (2014). Generalized outlier detection with flexible kernel density estimates. In Proceedings of the 2014 SIAM International Conference on Data Mining, pages 542–550. SIAM.

Sergin, N. D. and Yan, H. (2021). Toward a better monitoring statistic for profile monitoring via variational autoencoders. Journal of Quality Technology, 53(5):1–46.

Shi, C., Zhang, X., Sun, J., and Wang, L. (2022). Remote sensing scene image classification based on self-compensating convolution neural network. Remote Sensing, 14(3):545.

Shrestha, A. and Mahmood, A. (2019). Review of deep learning algorithms and architectures. IEEE Access, 7:53040–53065.

Smith, L. N. (2018). A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. arXiv preprint arXiv:1803.09820.

Stoumbos, Z. G., Jones, L. A., Woodall, W. H., and Reynolds, M. R. (2001). On nonparametric multivariate control charts based on data depth. In Frontiers in Statistical Quality Control 6, pages 207–227. Springer.

Struyf, A. J. and Rousseeuw, P. J. (1999). Halfspace depth and regression depth characterize the empirical distribution. Journal of Multivariate Analysis, 69(1):135–153.

Sun, Y., Ming, Y., Zhu, X., and Li, Y. (2022). Out-of-distribution detection with deep nearest neighbors. In International Conference on Machine Learning, pages 20827–20840. PMLR.

Tukey, J. W. (1975). Mathematics and the picturing of data. In Proceedings of the International Congress of Mathematicians, Vancouver, 1975, volume 2, pages 523–531.

Vakayil, A. and Joseph, V. R. (2022). Data twinning. Statistical Analysis and Data Mining: The ASA Data Science Journal.

Vencálek, O. (2017). Depth-based classification for multivariate data. Austrian Journal of Statistics, 46(3-4):117–128.

Villa-Pérez, M. E., Alvarez-Carmona, M. A., Loyola-Gonzalez, O., Medina-Pérez, M. A., Velazco-Rossell, J. C., and Choo, K.-K. R. (2021). Semi-supervised anomaly detection algorithms: A comparative summary and future research directions. Knowledge-Based Systems, 218:106878.

Voorhees, E. M. and Harman, D. (2000). Overview of the sixth text retrieval conference (TREC-6). Information Processing & Management, 36(1):3–35.

Wang, H. and Xia, Y. (2008). Sliced regression for dimension reduction. Journal of the American Statistical Association, 103(482):811–821.

Wang, X., Wang, Z., Shao, W., Jia, C., and Li, X. (2019). Explaining concept drift of deep learning models. In International Symposium on Cyberspace Safety and Security, pages 524–534. Springer.

Weese, M., Martinez, W., Megahed, F. M., and Jones-Farmer, L. A. (2016). Statistical learning methods applied to process monitoring: An overview and perspective. Journal of Quality Technology, 48(1):4–24.

Wu, H.-M. (2008). Kernel sliced inverse regression with applications to classification. Journal of Computational and Graphical Statistics, 17(3):590–610.

Yang, J., Wang, P., Zou, D., Zhou, Z., Ding, K., Peng, W., Wang, H., Chen, G., Li, B., Sun, Y., et al. (2022a). OpenOOD: Benchmarking generalized out-of-distribution detection. arXiv preprint arXiv:2210.07242.

Yang, J., Zhou, K., Li, Y., and Liu, Z. (2022b). Generalized out-of-distribution detection: A survey. arXiv preprint arXiv:2110.11334v2.

Yeganeh, A., Abbasi, S. A., Pourpanah, F., Shadman, A., Johannssen, A., and Chukhrova, N. (2022). An ensemble neural network framework for improving the detection ability of a base control chart in non-parametric profile monitoring. Expert Systems with Applications.

Yeung, D.-Y. and Chow, C. (2002). Parzen-window network intrusion detectors. In 2002 International Conference on Pattern Recognition, volume 4, pages 385–388. IEEE.

Yeung, D.-Y. and Ding, Y. (2003). Host-based intrusion detection using dynamic and static behavioral models. Pattern Recognition, 36(1):229–243.

Yin, Z. and Shen, Y. (2018). On the dimensionality of word embedding. Advances in Neural Information Processing Systems, 31.

Yu, F., Wang, D., Shelhamer, E., and Darrell, T. (2018). Deep layer aggregation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2403–2412.

Zan, T., Liu, Z., Su, Z., Wang, M., Gao, X., and Chen, D. (2020). Statistical process control with intelligence based on the deep learning model. Applied Sciences, 10(1):308.

Zhang, K., Bui, A. T., and Apley, D. W. (2023). Concept drift monitoring and diagnostics of supervised learning models via score vectors. Technometrics, 65(2):137–149.

Žliobaitė, I., Pechenizkiy, M., and Gama, J. (2016). An overview of concept drift applications. Big Data Analysis: New Algorithms for a New Society, pages 91–114.

Zou, C., Wang, Z., and Tsung, F. (2012). A spatial rank-based multivariate EWMA control chart. Naval Research Logistics (NRL), 59(2):91–110.

Zuo, Y. and Serfling, R. (2000). General notions of statistical depth function. Annals of Statistics, pages 461–482.

Appendix

A Addition to Experiment 1: Monte Carlo Study (Choice of Reference Samples)

To examine the influence of how the reference samples are formed, we conducted a Monte Carlo study with the data from Experiment 1. Here, we create reference samples for each class by randomly picking data points from correctly classified training data without considering confidence-related outcomes of the ANN.

The obtained results based on 10 runs are summarized in Table 4. Due to the extensive computational resources involved, we conduct the study for one type of Projection depth, namely for the symmetric case. For each choice of the data depth notion, the standard deviation does not exceed 0.01, meaning that the small number of iterations is sufficient for our investigation. The control charts based on **HD** achieve the highest *CDR* among all proposed control charts. Furthermore, the control charts based on **PD**$_2$ and **PD**$_3$ are more reliable during Phase II (In-control), resulting in a *SR* of 0.18.

In contrast to the results where the reference samples are selected according to the softmax scores, we observe low fluctuation in performance with the changing size $|R|$ and lower *CDR* values.

Evaluation	Size $|R|$	Phase	Observed process	Metric	MD	SD	HD$_r$	PD$_1$	PD$_2$	PD$_3$
	2000	I	In-control	FAR	0.05	/	0.05	0.05	0.05	0.05
	2000	II	In-control	SR	0.21	/	0.23	0.19	0.18	0.18
	2000	II	Out-of-control	CDR	0.62	/	0.64	0.58	0.59	0.59
	3000	I	In-control	FAR	0.05	/	0.05	0.05	0.05	0.05
Experiment 1	3000	II	In-control	SR	0.21	/	0.23	0.20	0.18	0.18
$\boldsymbol{m}_i \in \mathbb{R}^{16}$	3000	II	Out-of-control	CDR	0.62	/	0.64	0.59	0.59	0.59
	4000	I	In-control	FAR	0.05	/	0.05	0.05	0.05	0.05
	4000	II	In-control	SR	0.21	/	0.23	0.20	0.18	0.18
	4000	II	Out-of-control	CDR	0.62	/	0.65	0.60	0.60	0.59

Table 4: Monte Carlo study: Performance of r control charts ($\alpha = 0.05$) in Experiment 1 with reference samples R being predicted classes that were randomly constructed. The underlined numbers indicate the suggested method based on the trade-off between *SR* and *CDR*. We do not compute **SD** due to the computational complexity being $O(n^{k+1})$.

Thus, we recommend choosing the reference sample based on the intended purpose of the monitoring. When accepting higher signal rates in the in-control phase (potentially due to misclassification), reference samples should be chosen based on the softmax scores. In turn, this leads to more sensitive detection of out-of-control samples.

B Experiment 2: Multiclass Classification of Questions

For the second experiment, we use the Text REtrieval Conference (TREC) dataset which consists of fact-based questions divided into six broad semantic categories[2] (cf. Voorhees and Harman, 2000). The model was trained with the four classes: "Numeric values", "Description and abstract concepts", "Entities" and "Human beings". Examples of such questions can be found in Table 5. The classification task is to assign an incoming question to one of four categories.

The trained neural network contains three hidden layers: a word embedding layer, a Long Short-Term Memory (LSTM) layer, and a fully connected layer which we use as the embedding generator of the size 1×8. After that, the output layer returns softmax vector 1×4, where the maximum value corresponds to the label of the predicted category. It is worth noting that here the word embedding layer is not a part of our monitoring approach but a Natural Language Processing (NLP) technique that enables the model to associate a numerical vector to every word so that the distance between any two vectors is related to the semantic meaning of the encrypted words (cf. Yin and Shen, 2018). For the interested reader, we recommend referring to publications that offer a comprehensive introduction

---

[2]https://cogcomp.seas.upenn.edu/Data/QA/QC/

Class	Example
Numeric values	What is the size of Argentina?
Description and abstract concepts	What is artificial intelligence?
Entities	What is the tallest piece on a chessboard?
Human beings	Who invented basketball?

Table 5: Question examples and respective four categories of the TREC dataset used for training the ANN in Experiment 2.

to neural networks for NLP tasks, for example, Goldberg (2016) and Nammous and Saeed (2019).

In total, 700 data points of each category were used as the training data. The achieved accuracy on the test dataset that contains 150 unseen samples for each class is 81.17% after 25 training epochs. The 60 out-of-control samples were taken from two other semantic categories that were not used for training, namely "Abbreviations" and "Locations".

B.1 Monitoring Results

Considering the results of Experiment 2 in Table 6, we notice a decreasing *SR* when the size of the reference sample is increasing. Nevertheless, the *SR* values remain substantially high for considered control charts. Although the further increase of reference samples might improve monitoring during the in-control state, it negatively affects the detection of anomalous data. As we can observe in the case of $\mathbf{HD}_r$, the *CDR* is reduced by 25%, changing from $|R| = 400$ to $|R| = 600$.

Overall, we notice that either a paired control chart or another procedure to decide confidently when the data points are in- our out-of-control is required to improve the performance in Experiment 2. However, to understand what could be the underlying reasons for unsatisfactory results, we inspect the misclassification effect as well as visually compare the in- and out-of-control data.

B.2 Misclassification and Data Diagnostic

There are 113 samples from the test data that were misclassified, and the softmax output serves as a model's confidence about its prediction in our case. Looking at the density plots in Figure 6, we notice an evident difference in distributions between correctly classified and misclassified samples in Phase II (in-control). Remarkably, the score of out-of-control samples are rather high and resemble the

Evaluation	Size $\mid R \mid$	Phase	Observed process	Metric	**MD**	**SD**	**HD$_r$**	**PD$_1^a$**	**PD$_2^a$**	**PD$_3^a$**	**PD$_1$**	**PD$_2$**	**PD$_3$**
Experiment 2 $\boldsymbol{m}_i \in \mathbb{R}^8$	400	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	400	II	In-control	*SR*	0.68	/	0.69	0.64	0.62	0.64	0.65	0.65	0.66
	400	II	Out-of-control	*CDR*	0.58	/	0.67	0.48	0.50	0.50	0.52	0.53	0.55
	500	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	500	II	In-control	*SR*	0.45	/	0.60	0.45	0.43	0.44	0.44	0.45	0.46
	500	II	Out-of-control	*CDR*	0.45	/	0.58	0.43	0.37	0.37	0.40	0.38	0.45
	600	I	In-control	*FAR*	0.05	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	600	II	In-control	*SR*	0.35	/	0.37	0.34	0.34	0.33	0.32	0.32	0.33
	600	II	Out-of-control	*CDR*	0.30	/	0.42	0.32	0.30	0.30	0.30	0.30	0.30
Experiment 3 $\boldsymbol{m}_i \in \mathbb{R}^3$	50	I	In-control	*FAR*	0.04	0.00	0.04	0.04	0.04	0.04	0.04	0.04	0.04
	50	II	In-control	*SR*	0.26	0.00	0.61	0.16	0.16	0.05	0.00	0.05	0.05
	50	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	60	I	In-control	*FAR*	0.05	0.00	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	60	II	In-control	*SR*	0.16	0.00	0.50	0.26	0.24	0.16	0.00	0.00	0.00
	60	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	70	I	In-control	*FAR*	0.04	0.00	0.04	0.04	0.04	0.04	0.04	0.04	0.04
	70	II	In-control	*SR*	0.05	0.00	0.37	0.00	0.05	0.03	0.11	0.11	0.11
	70	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Table 6: Performance of *r* control charts ($\alpha = 0.05$) in the presented experiments with *R* being the predicted class. Green rows highlight Phase I false alarm rates (*FAR*), and violet define Phase II signal and correct detection rates (*SR* and *CDR*), respectively. The underlined numbers indicate the suggested method based on the trade-off between *SR* and *CDR*. We compute **SD** for $\mathbb{R}^3$ only, as computational complexity is $O(n^{k+1})$.

scores of the in-control samples from Phase II. This behavior can be explained by the similarity of the out-of-control data and the reference samples. More precisely, the network is trained to distinguish the phrases based on features that are identical for both the in-control and out-of-control samples. However, it is worth noting that this could be different on lower aggregation levels (i.e., hidden layers closer to the input).

The Radial Coordinate visualization (Radviz) shown in Figure 7 reveals that this issue is present in our case: The out-of-control data often overlap with the reference samples. Moreover, most of the in-control (test data) regions are not covered by the reference samples. However, to answer the question of whether the high signal rate is partially due to the misclassification samples, we compute conditional sample rates on misclassification (*SR*|*M*) and correct prediction (*SR*|*C*).

As shown in Table 7, the additional information about misclassified samples could considerably improve the *SR* results (cf. the outcome for $\mid R \mid = 600$ of *SR*|*C*). Moreover, we notice that the majority of the misclassified samples would be flagged as anomalous, with the signal rates declining when the reference sample size grows. Hence, the combination of our monitoring approach with an additional misclassification detection technique could lead to a more reliable nonstationarity detection.

Fig. 6: Kernel density estimates of the score distributions in Phase II for in-control (test data) and out-of-control samples.

Evaluation	Size $|R|$	Metric	**MD**	**SD**	**HD**$_r$	**PD**$_1^a$	**PD**$_2^a$	**PD**$_3^a$	**PD**$_1$	**PD**$_2$	**PD**$_3$
	400	*SR|M*	0.86	/	0.87	0.81	0.82	0.84	0.83	0.86	0.86
Experiment 2	400	*SR|C*	0.63	/	0.65	0.60	0.57	0.60	0.61	0.60	0.61
	500	*SR|M*	0.80	/	0.84	0.79	0.76	0.76	0.76	0.78	0.79
$m_i \in \mathbb{R}^8$	500	*SR|C*	0.37	/	0.55	0.37	0.35	0.36	0.37	0.37	0.38
	600	*SR|M*	0.69	/	0.72	0.64	0.65	0.61	0.65	0.64	0.64
	600	*SR|C*	0.28	/	0.29	0.26	0.26	0.26	0.25	0.25	0.26

Table 7: Summary of signal rates under the condition that the samples were misclassified (*SR\|M*) or correctly classified (*SR\|C*) in Phase II. We do not compute **SD** due to the computational complexity being $O(n^{k+1})$.

Fig. 7: Visualization of the Reference Samples (*RS*) $\{R_1, \ldots, R_4\}$ with $|R| = 400$ and the data from Phase II, in-control part (test data) from Experiment 2. $V_1, \ldots, V_8$ define anchors which correspond to neurons that produced embeddings $m_i \in \mathbb{R}^8$. Density contour plots outline respective classes.

C   Experiment 3: Binary Classification of Sonar Signals

In the third experiment, we consider a binary classification problem of sonar data (Dua and Graff, 2019). This dataset summarizes sonar signals collected from metal cylinders and cylindrically shaped rocks (Gorman and Sejnowski, 1988). There are 208 samples in total, comprising 111 metal cylinders and 97 rock returns. Each sample consists of a series of 60 numbers ranging from 0.0 to 1.0, representing a normalized spectral envelope. The task of the classifier is to distinguish which samples are from scanning a rock and which are from a metal cylinder.

Our model is an FNN with the architecture $60 \to 30 \to 15 \to 3 \to 1$ that comprises four fully connected layers reducing the complexity 1×60 of the input data by first processing it through the hidden layers that have 30 and 15 neurons. Afterward, the compressed data representation enters the layer with 3 neurons whose output is also used as embeddings of size 1×3 for the monitoring procedure. Then, to obtain the class label, we transform the interim output from size 1×3 to 1×1. As we have a binary classification problem, we use only one neuron in the output layer together with the sigmoid activation function that is centered around 0.5, returning the probability of the processed sample belonging to class 2. Thus, if the result of the output layer is 0.5 or higher, we conclude that the processed sample is a part of class 2 (rock) and of class 1 (metal cylinder) otherwise. Due to the small size of the dataset, only 38 samples (24 of class 1 and 14 of class 2) are allocated to the testing stage which is later used in Phase II as the in-control data. Consequently, the remaining 85 metal cylinders and 85 rock examples are taken for training the FNN. The performance metrics such as validation loss and accuracy are used to determine the number of epochs, i.e., training cycles in which the model learns from the data and updates the parameters. Following that, the FNN model was trained for 39 epochs and achieved 81.58% accuracy on the test data.

To create out-of-control samples, by flattening the input we estimate the parameters of a beta distribution for each class (i.e., $\tilde{\alpha}_1$, β_1 of class 1, and $\tilde{\alpha}_2$, β_2 of class 2) and randomly sample from a beta distribution with parameters $\tilde{\alpha}_v = \tilde{\alpha}_1/\tilde{\alpha}_2$ and $\beta_v = \beta_1/\beta_2$ to generate 30 out of control observations.

Studying the outcomes of Phase I in Experiment 3, Table 6 we notice that although we choose $\alpha = 0.05$, *FAR* sometimes equals 0.04. This happens due to rounding, thus, *FAR* = 0.04 for $|R| = 50$ and $|R| = 70$ coincides with the expected value. Consequently, the expected value of *FAR* is reached for each notion of data depth except Simplicial depth. The reason for that is the large number of data

Fig. 8: Reference samples ($|R| = 70$) and out-of-control embeddings from Experiment 3. Blue-colored points belong to class 1, green-colored to class 2, and magenta-colored are out-of-control samples. The size of the points is proportional to the value of Simplicial depth and the gray-colored points highlight the data samples that received $\mathbf{SD}(\boldsymbol{m}_t) = 0$.

points in the reference sample with $\mathbf{SD}(\boldsymbol{m}_t) = 0$. That can be explained by Simplicial depth assigning zero to every point in the space that lies outside the convex hull of the sample (Afshani et al., 2016; Francisci et al., 2019). In Figure 8, we observe the dispersion of the data and how many samples obtained $\mathbf{SD}(\boldsymbol{m}_t) = 0$. All out-of-control samples received the predictions of class 1, meaning that their depth is determined with respect to the blue point cloud. Overall, the control charts operate well with $|R| = 50$ and the symmetric projection depth, especially with $\mathbf{PD}_2$ and $\mathbf{PD}_3$. Alternatively, one can choose $\mathbf{MD}$ with $|R| = 70$ because of correctly reached *SR* and similarly high *CDR*.

D   The Q Control Chart (Batch Size > 1)

Similarly to the *r* control chart, the Q control chart proposed by Liu (1995) is based on ranks of multivariate observations which are obtained by computing data depth. The test statistic of the Q control chart is the average of consecutive subsets of $r^c_.(\boldsymbol{m}_i)$, being

$$Q^c_.(\boldsymbol{m}_i) = \frac{1}{n} \sum_{j=1}^{n} r^c_.(\boldsymbol{m}_{ij})$$

41

with the batch size n (Liu, 1995). The interpretation of the ranks in the Q control chart is similar to the interpretation in the case of the r control chart.

To compute the control limit, we use the equation

$$LCL = \frac{(n!\alpha)^{1/n}}{n},$$

given that $\alpha \leq \frac{1}{n!}$ (Stoumbos et al., 2001). However, if $\alpha > \frac{1}{n!}$, the LCL has to be computed numerically by solving the polynomial equation provided by Liu (1995). In general, the process is considered to be out of control if $Q_.^c(\mathbf{m}_i) \leq LCL$.

To evaluate the performance of Q control charts, we chose those $|R|$ which achieved the most satisfactory (trade-off) performance with r control charts. Due to the small size of the dataset in Experiment 3, we compute the Q control charts with the batch size $n = 5$ only for Experiments 1 and 2.

Considering the results in Table 8, we notice that (apart from **SD**) the SR values are excessively high. The increase can be explained by a substantial change in control limits. Referring to the previous description, for $n = 3$, we obtain $LCL = 0.22$, and for $n = 5$, $LCL = 0.29$. At the same time, in all possible consolidations, the Q control chart achieves notable results during the out-of-control period, considerably improving the CDR values in Experiment 2 for Projection depth. Thus, if a supplementary procedure can be developed for detecting and filtering signals successfully when the process remains in control, the Q control chart would outperform the r control chart.

E Reference Sample in Form of Merged Classes

In this analytical part, we monitor the data in each of the three experiments by creating the reference samples without conditioning on the (predicted) class c. That is, the Phase II samples are compared to a joint embedding distribution from Phase I, being independent of the class labels. The benefit of this approach is that no predictions are needed, meaning that if the ANN model provides an incorrect class label, the possible negative effect of misclassification is excluded. Additionally, the application of merged reference samples implies that if a data point is flagged as out of control, it would remain out of control compared to the entire reference data.

Reporting one case for each experiment in Table 9, we can observe satisfactory performance in Experiment 3. In Experiments 1 and 2, the results are less convincing during the out-of-control part.

Evaluation	Batch size	Phase	Observed process	Metric	MD	SD	HD$_r$	PD$_1^a$	PD$_2^a$	PD$_3^a$	PD$_1$	PD$_2$	PD$_3$
	3	I	In-control	FAR	0.08	/	0.09	<u>0.07</u>	0.08	0.08	0.08	0.08	0.08
Experiment 1	3	II	In-control	SR	0.58	/	0.59	<u>0.55</u>	0.56	0.56	0.55	0.54	0.55
$\boldsymbol{m}_i \in \mathbb{R}^{16}$	3	II	Out-of-control	CDR	0.98	/	0.98	<u>0.98</u>	0.97	0.97	0.98	0.96	0.97
$\|R\| = 3000$	5	I	In-control	FAR	0.10	/	0.13	0.10	0.10	0.11	<u>0.10</u>	<u>0.10</u>	0.11
	5	II	In-control	SR	0.75	/	0.76	0.72	0.73	0.73	<u>0.71</u>	<u>0.71</u>	0.71
	5	II	Out-of-control	CDR	1.00	/	1.00	1.00	1.00	1.00	<u>1.00</u>	<u>1.00</u>	1.00
	3	I	In-control	FAR	0.15	/	<u>0.16</u>	0.13	0.11	0.11	0.15	0.14	0.14
Experiment 2	3	II	In-control	SR	0.73	/	<u>0.75</u>	0.72	0.71	0.72	0.73	0.72	0.72
$\boldsymbol{m}_i \in \mathbb{R}^{8}$	3	II	Out-of-control	CDR	0.67	/	<u>0.72</u>	0.62	0.57	0.62	0.67	0.57	0.72
$\|R\| = 400$	5	I	In-control	FAR	0.19	/	<u>0.19</u>	0.17	0.15	0.16	0.20	0.18	0.18
	5	II	In-control	SR	0.74	/	<u>0.77</u>	0.74	0.74	0.74	0.74	0.75	0.74
	5	II	Out-of-control	CDR	0.80	/	<u>0.90</u>	0.80	0.70	0.80	0.80	0.80	0.80
Experiment 3	3	I	In-control	FAR	0.07	0.00	0.11	<u>0.02</u>	<u>0.02</u>	<u>0.02</u>	0.04	0.04	0.04
$\boldsymbol{m}_i \in \mathbb{R}^{3}$	3	II	In-control	SR	0.36	0.00	0.34	<u>0.18</u>	<u>0.18</u>	<u>0.18</u>	0.36	0.36	0.27
$\|R\| = 70$	3	II	Out-of-control	CDR	1.00	0.00	1.00	<u>1.00</u>	<u>1.00</u>	<u>1.00</u>	1.00	1.00	1.00

Table 8: Performance of Q control charts ($LCL = 0.22$ for $n = 3$ and $LCL = 0.29$ for $n = 5$) in the presented experiments with R being the predicted class. The underlined numbers indicate the suggested method based on the trade-off between *SR* and *CDR*. We compute **SD** for $\mathbb{R}^3$ only, as computational complexity is $O(n^{k+1})$.

The reason is that the data depth values of reference sample points in a merged case are considerably lower than in a case of individual classes, leading to a less sensitive detection of nonstationary samples. On the contrary, the *SR* values are reduced compared to the case when the reference sample relates to the predicted class only.

Although the calculation of the data depth with respect to a merged version of a reference sample eliminates the misclassification problem, in our empirical study the detection results of spurious data by using predicted classes on their own are substantially better for high-dimensional problems. For low-dimensional cases such as Experiment 3, we recommend first examining the performance of the monitoring based on the merged reference sample of different sizes, and then, if it is not operating acceptably, applying the method with the reference samples of predicted classes.

F Computation Time

For performing online surveillance, the computation time of the monitoring statistic is of particular importance. As the most time-consuming part of our approach is the derivation of data depth values, we compare the execution time of different algorithms to obtain the depth of one data point. To provide a concise summary, we compare running times for the middle sizes of reference samples, namely $\|R\| = \{3000, 500, 60\}$, and for the cases displayed in Table 9. In the case of Projection depth,

Evaluation	Size $\|R\|$	Phase	Observed process	Metric	**MD**	**SD**	**HD_r**	**PD_1^a**	**PD_2^a**	**PD_3^a**	**PD_1**	**PD_2**	**PD_3**
	30000	I	In-control	*FAR*	0.05	/	<u>0.05</u>	0.05	0.05	0.05	0.05	0.05	0.05
Experiment 1	30000	II	In-control	*SR*	0.15	/	<u>0.20</u>	0.10	0.09	0.08	0.09	0.09	0.09
$m_i \in \mathbb{R}^{16}$	30000	II	Out-of-control	*CDR*	0.19	/	<u>0.24</u>	0.02	0.02	0.01	0.02	0.02	0.02
	1600	I	In-control	*FAR*	<u>0.05</u>	/	0.05	0.05	0.05	0.05	0.05	0.05	0.05
Experiment 2	1600	II	In-control	*SR*	<u>0.66</u>	/	0.08	0.02	0.00	0.00	0.02	0.08	0.00
$m_i \in \mathbb{R}^8$	1600	II	Out-of-control	*CDR*	<u>0.63</u>	/	0.05	0.00	0.00	0.00	0.02	0.02	0.02
	140	I	In-control	*FAR*	0.05	0.00	0.05	0.05	<u>0.05</u>	<u>0.05</u>	0.05	0.05	0.05
Experiment 3	140	II	In-control	*SR*	0.08	0.00	0.00	0.03	<u>0.05</u>	<u>0.05</u>	0.08	0.08	0.08
$m_i \in \mathbb{R}^3$	140	II	Out-of-control	*CDR*	1.00	0.00	1.00	1.00	<u>1.00</u>	<u>1.00</u>	1.00	1.00	1.00

Table 9: Performance of *r* control charts ($\alpha = 0.05$) in the presented experiments with *R* being merged classes. The underlined numbers indicate the suggested method based on the trade-off between *SR* and *CDR*. We compute **SD** for $\mathbb{R}^3$ only, as computational complexity is $O(n^{k+1})$.

we present the results of the symmetric type with three applied algorithms.

In Figure 11, we can see that Simplicial depth requires considerably longer to be computed than other notions of data depth. Regarding the algorithms to approximate Projection depth, the running times remain similar in Experiments 1 and 2, Figures 9 and 10, respectively. Despite the increased complexity of experiments, Mahalanobis depth is characterized by a stable and low running time. On the contrary, the running time of **HD_r** increases noticeably with the growing size of reference samples as well as additional dimensions.

To summarize, if we exclude the performance of **MD**, the computation of data depth in $\mathbb{R}^{16}$ for one point with $\|R\| = 3000$ would usually take more than 10 seconds. In statistics, such results seem to be acceptable. However, taking into account the current applications of ANNs, for example, the image classification applying a CNN, the time required to process one image is under 0.1 second (cf. Shi et al., 2022). Thus, we should critically consider the running times for the computation of data depth, striving for their improvements and guarantee the applicability of the proposed framework to monitor state-of-the-art models based on AI by improving the software for data depth computation.

(a) Single class reference samples of size $|R| = 3000$ **(b)** Merged reference samples of size $|R| = 30000$

Fig. 9: Distribution of computation time for different data depths in Experiment 1. The order is symmetric Projection Depth with **C**oordinate **D**escent, **N**elder-**M**ead and **R**efined **R**andom algorithms, **R**obust **H**alfspace and **M**ahalanobis **D**epths.

(a) Single class reference samples of size $|R| = 500$

(b) Merged reference samples of size $|R| = 1600$

Fig. 10: Distribution of computation time for different data depths in Experiment 2 on a logarithmic scale. The order is symmetric Projection Depth with **C**oordinate **D**escent, **N**elder-**M**ead and **R**efined **R**andom algorithms, **R**obust **H**alfspace and **M**ahalanobis **D**epths.

(a) Single class reference samples of size $|R| = 60$

(b) Merged reference samples of size $|R| = 140$

Fig. 11: Distribution of computation time for different data depths in Experiment 3 on a logarithmic scale. The order is symmetric Projection Depth with **C**oordinate **D**escent, **N**elder-**M**ead and **R**efined **R**andom algorithms, **R**obust **H**alfspace, **S**implicial and **M**ahalanobis **D**epths.