Information Sciences 564 (2021) 124–143

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Blessing of dimensionality at the edge and geometry of few-shot learning

Ivan Y. Tyukin [a,c,f,*], Alexander N. Gorban [a,b], Alistair A. McEwan [a,g], Sepehr Meshkinfamfard [d], Lixin Tang [e]

[a] University of Leicester, UK
[b] Lobachevsky University, Russia
[c] St Petersburg State Electrotechnical University, Russia
[d] University College London, UK
[e] Key Laboratory of Data Analytics and Optimization for Smart Industry (Northeastern University), Ministry of Education, People's Republic of China
[f] Norwegian University of Science and Technology, Norway
[g] University of Derby, UK

ARTICLE INFO

Article history:
Received 3 October 2019
Received in revised form 17 December 2020
Accepted 8 January 2021
Available online 3 February 2021

Keywords:
Stochastic separation theorems
Artificial intelligence
Machine learning
Computer vision

ABSTRACT

In this paper we present theory and algorithms enabling classes of Artificial Intelligence (AI) systems to continuously and incrementally improve with a priori quantifiable guarantees – or more specifically remove classification errors – over time. This is distinct from state-of-the-art machine learning, AI, and software approaches. The theory enables building few-shot AI correction algorithms and provides conditions justifying their successful application. Another feature of this approach is that, in the supervised setting, the computational complexity of training is linear in the number of training samples. At the time of classification, the computational complexity is bounded by few inner product calculations. Moreover, the implementation is shown to be very scalable. This makes it viable for deployment in applications where computational power and memory are limited, such as embedded environments. It enables the possibility for fast on-line optimisation using improved training samples. The approach is based on the concentration of measure effects and stochastic separation theorems and is illustrated with an example on the identification faulty processes in Computer Numerical Control (CNC) milling and with a case study on adaptive removal of false positives in an industrial video surveillance and analytics system.

1. Introduction

The past decade has seen extraordinary growth and advances in technologies for collecting and processing very large data streams and data sets. Central to these advances has been the area of Artificial Intelligence (AI) built on Machine Learning (ML) and Data Analytics theories. Exploitation of AI is becoming overwhelmingly ubiquitous. For instance, end users and consumers use mobile phone apps with AI capabilities, security systems may employ AI to identify unwanted intrusions and infringements, healthcare systems may use AI to assist clinical diagnosis or processes, and mechatronic systems may use AI to implement control including autonomous and semi-autonomous functionality. Examples of these in literature

* Corresponding author at: University of Leicester, Department of Mathematics, University Road, LE1 7RH, UK.
E-mail addresses: I.Tyukin@le.ac.uk (I.Y. Tyukin), A.N.Gorban@le.ac.uk (A.N. Gorban), A.McEwan@derby.ac.uk (A.A. McEwan), s.meshkinfamfard@ucl.ac.uk (S. Meshkinfamfard), lixintang@ise.neu.edu.cn (L. Tang).

https://doi.org/10.1016/j.ins.2021.01.022
0020-0255/

include those reported in [32,46]. Whilst this increase in application areas is due in part to advances in AI and ML, it is also due to advances in hardware and supporting platforms. The emergence of devices such as Nvidia GPUs and Google TPUs have meant that power of server- or super-computer platforms are no longer necessarily required for deployment of deep learning systems.

Whilst state-of-the-art AI systems may be capable of outperforming both human and other data mining approaches in identifying minute patterns in very large data sets, their conclusions are vulnerable to data inconsistencies, poor data quality, and the uncertainty inherent to any data. This uncertainty, together with engineering constraints on implementation and systems integration, leads to inevitable and unavoidable errors.

Consequences of errors resulting from AI may range from minor inconveniences to safety–critical risks: incorrect cancer treatment options [44] and crashes of autonomous vehicles are a few examples of the latter (see [3] and references therein for further detail and examples). However the solution to ameliorating or eliminating errors is non-trivial. The field of Software Engineering has provided numerous approaches for understanding the behaviours and misbehaviours of software based systems ranging from efficient scenario based testing techniques through to formal verification—see for instance [19]. However these software architectures typically do not contain the inherent uncertainties of data driven AI—although this is rapidly changing with the push towards higher levels of driver assist and autonomy. Recent examples that consider autonomous vehicle control systems incorporating AI include [31,34], although common to these approaches is to look at systems level behaviours rather than the correctness of the AI component.

Whilst structuring data, improving the quality of data, and removing uncertainty is known to improve quality, it is too resource intensive in the general case and thus unsustainable across sectors and industries. Moreover, whilst it may improve the quality of high-assurance or safety–critical systems, it does not provide a measure of understanding of quality and consistency of output. More fundamentally, constraints on implementations such as quantization errors and memory limits present challenges to AI performance in resource constrained embedded settings—referred to as ‘‘at the edge” or ‘‘edge-based”.

1.1. Background and related work

Significant efforts have been applied to address errors in AI systems. Using ensembles [20,23,24], augmenting training data [29,36,38], enforcing continuity [56], and AI knowledge transfer [37,6,50] have been extensively discussed in the literature. These measures, however, do not warrant error-free behaviour as AIs based on empirical data are expected to make mistakes.

Recently in [11,8,48,12,13,47] we have shown that spurious errors of AI systems operating in high-dimensional spaces (convolutional and deep learning neural networks being the canonical examples) can be efficiently removed by Fisher discriminants. The advantage of this approach over, for instance, Support Vector Machines (SVM) [49] is that the computational complexity for constructing Fisher discriminants is at most linear in the number of points in the training set whereas the worst-case complexity for SVM scales as a cubic function of the training set size [5].

This method is applicable to identified singular spurious errors as well as to moderate-sized clusters. An open question is what happens when the volume of errors grows large and becomes comparable or even exceeds the volume of correct responses? Is it possible for a deployed AI to keep improving its performance with limited resources available for supervision and re-training by learning from new examples as they arrive? Both of these questions are fundamentally relevant across the spectrum of AI applications. Notably, given the computational and data management costs needed for continuous complete re-training, these questions are particularly acute for edge-based systems.

1.2. Contribution and structure of this paper

In this paper we show that stochastic separation theorems, or the blessing of dimensionality [9,26], stemming from the concentration of measure effects [10,25,18], can be adapted and applied to address these questions. We present and justify both mathematically and experimentally an algorithm capable of delivering the removal of errors at computational costs compatible with deployment at the edge. The algorithm has both supervised and unsupervised components which enables it to adapt to data without additional supervisory inputs. As compared to previously proposed stochastic separation-based algorithms [48,47], current algorithm was shown to consistently learn from large volumes of errors making its deployment in applications with uncertainty and bias in training data particularly attractive.

The paper is organized as follows. Section 2 sets out the notation we use in this paper. Section 3 contains necessary theoretical preliminaries and formal statement of the problem. In Section 4 we present a new algorithm for improving AIs ‘‘at the edge”. Section 5 discusses and interprets our results in the context of existing work in the area of few-shot learning. Section 6 illustrates an application of the proposed algorithms in two industrial applications: product quality prediction in milling machines and automated edge-based object detection in large-scale survaliance systems; Section 7 concludes the paper.

2. Notation

The following notational agreements are used throughout the text:

- $\mathbb{R}^n$ stands for the n-dimensional linear real vector space, and $\mathbb{R}_{\geqslant 0} = \{x \in \mathbb{R} |\ x \geqslant 0\}$;
- $\mathbb{N}$ denotes the set of natural numbers;
- symbols $\boldsymbol{x} = (x_1, \ldots, x_n)$ will denote elements of $\mathbb{R}^n$;
- $(\boldsymbol{x}, \boldsymbol{y}) = \sum_k x_k y_k$ is the inner product of $\boldsymbol{x}$ and $\boldsymbol{y}$, and $\|\boldsymbol{x}\| = \sqrt{(\boldsymbol{x}, \boldsymbol{x})}$ is the standard Euclidean norm in $\mathbb{R}^n$;
- $\mathbb{B}_n(r, \boldsymbol{y})$ stands for the ball in $\mathbb{R}^n$ of radius r centered at $\boldsymbol{y}$: $\mathbb{B}_n(r, \boldsymbol{y}) = \{\boldsymbol{x} \in \mathbb{R}^n |\ (\boldsymbol{x} - \boldsymbol{y}, \boldsymbol{x} - \boldsymbol{y}) \leqslant r^2\}$;
- $\mathbb{B}_n$ denotes for the unit ball in $\mathbb{R}^n$ centered at the origin: $\mathbb{B}_n = \{\boldsymbol{x} \in \mathbb{R}^n |\ (\boldsymbol{x}, \boldsymbol{x}) \leqslant 1\}$;
- V_n is the n-dimensional Lebesgue measure, and $V_n(\mathbb{B}_n)$ is the volume of unit ball;
- if $\mathcal{Y}$ is a finite set then the number of elements in $\mathcal{Y}$ (cardinality of $\mathcal{Y}$) is denoted by $|\mathcal{Y}|$.
- if $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_k$ are elements of $\mathbb{R}^n$ then $\mathrm{Conv}(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$ denotes the convex hull of $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k$:

$$\mathrm{Conv}(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k) = \left\{ \boldsymbol{y} \in \mathbb{R}^n | \boldsymbol{y} = \sum_{i=1}^{k} \lambda_i \boldsymbol{y}_i,\ \lambda_i \geqslant 0, \sum_{i=1}^{k} \lambda_i = 1 \right\}.$$

3. Problem formulation and mathematical preliminaries

3.1. Problem formulation

Following [13], we consider a generic AI system that processes some *input* signals, produces *internal* representations of the input and returns some *outputs*. We assume that there is a sampling process whereby some relevant information about the input, internal signals, and outputs are combined into a common vector, $\boldsymbol{x}$, representing, but not necessarily defining, the *state* of the AI system.

Depending on the sampling process, the vector $\boldsymbol{x}$ may have various numbers of elements. But generally, the objects $\boldsymbol{x}$ are assumed to be elements of $\mathbb{R}^n$, with n depending on the sampling process. Over a period of activity the AI system generates a set $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ of representations $\boldsymbol{x}$. In agreement with standard assumptions in machine learning literature [49], we assume that the set $\mathcal{X}$ is a random sample drawn from some distribution. The distribution that generates vectors $\boldsymbol{x}$ is supposed to be *unknown*. We will, however, impose some mild technical assumption on the generating probability distribution.

The central question we would like to address here is how to create algorithms capable of producing a single or an esemple of linear functionals suitable for decision-making at-the-edge. The focus on linear functionals is motivated by computational efficiency of their implementation in embedded settings. We would like to avoid using the framework of Support Vector Machines due to the computational costs involved which may present an obstable for embedded deployment.

Assumption 1. The probability density function, p, associated with the probability distribution of the random variable $\boldsymbol{x}$ exists, is defined on the unit ball $\mathbb{B}_n$, and there exist $C > 0$ and $r \in (0, 2)$ such that

$$\int_{\mathbb{B}_n(1/2, \boldsymbol{z})} p(\boldsymbol{x}) d\boldsymbol{x} \leqslant C \left(\frac{r}{2}\right)^n \text{ for all } \boldsymbol{z} \in \mathbb{B}_n,\ \|\boldsymbol{z}\| = 1/2. \tag{1}$$

The assumption requires that the vector-valued random variable $\boldsymbol{x}$ is in $\mathbb{B}_n$ which is consistent with the scope of our applications. The other part of the assumption, condition (1), is a version of the Smeared Absolute Continuity (SmAC) property introduced in [12,14]. Awareness of the latter property will be important for the algorithms that follow. In addition to Assumption 1 it will be convenient to consider alternative specifications of the data probability distribution which are captured in Assumption 2 below.

Assumption 2. The probability density function, p, associated with the probability distribution of the random variable $\boldsymbol{x}$ exists, is defined on the unit ball $\mathbb{B}_n$, and there exist $C > 0$ and $r \in \mathbb{R}_{\geqslant 0}$ such that

$$p(\boldsymbol{x}) < \frac{C r^n}{V_n(\mathbb{B}_n)}, \tag{2}$$

for all $\boldsymbol{x} \in \mathbb{B}_n$.

Note that if Assumption 2 holds with $r \in (0, 2)$ then it automatically implies that Assumption 1 holds true too. In case of Assumption 2 we do not wish yet to specify exact values of C and r. We would, however, like to formalise a particularly useful form of its dependence on dimension n and the volume of the unit ball $\mathbb{B}_n$ captured by (2).

Definition 1. *A point $\boldsymbol{x} \in \mathbb{R}^n$ is linearly separable* from a set $\mathcal{Y} \subset \mathbb{R}^n$, if there exists a linear functional $l(\cdot)$ such that

$$l(\boldsymbol{x}) > l(\boldsymbol{y})$$

for all $\boldsymbol{y} \in \mathcal{Y}$.

Definition 2. *A set $\mathcal{X} \subset \mathbb{R}^n$ is linearly separable from a set $\mathcal{Y} \subset \mathbb{R}^n$*, if there exists a linear functional $l(\cdot)$ such that

$$l(\boldsymbol{x}) > l(\boldsymbol{y})$$

for all $\boldsymbol{y} \in \mathcal{Y}$ and $\boldsymbol{x} \in \mathcal{X}$.

In addition to these standard notions of linear separability, we adopt the notion of Fisher separability [12,14]. Observe that the classical Fisher separability requires Mahalanobis inner product or whitening. Hereinafter, we assume that the approximate whitening and centralization for the dataset are completed and the inner product is fixed.

Definition 3. *A point $\boldsymbol{x} \in \mathbb{R}^n$ is Fisher separable* from a set $\mathcal{Y} \subset \mathbb{R}^n$, if

$$(\boldsymbol{x}, \boldsymbol{x}) > (\boldsymbol{x}, \boldsymbol{y}) \tag{3}$$

for all $\boldsymbol{y} \in \mathcal{Y}$. The point is Fisher separable from the set $\mathcal{Y}$ with a threshold $\kappa \in [0, 1)$ if

$$(\boldsymbol{x}, \boldsymbol{x}) > \kappa(\boldsymbol{x}, \boldsymbol{y}) \tag{4}$$

Having introduced all relevant assumptions and notions, we are now ready to proceed with results underpinning our algorithmic developments.

3.2. Mathematical preliminaries

Our first result is provided in Theorem 1 (cf. [15,12]) which is a prototype for a large family of stochastic separation theorems [17].

Theorem 1. *Let $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ be given, $\boldsymbol{x}_i \in \mathbb{B}_n$, and let $\boldsymbol{x}$ be drawn from a distribution satisfying Assumption 1. Then $\boldsymbol{x}$ is Fisher separable from the set $\mathcal{X}$ with probability*

$$P \geqslant 1 - MC\left(\frac{r}{2}\right)^n, \ r \in (0, 2). \tag{5}$$

Proof of Theorem 1. Consider events

$$A_i : \boldsymbol{x} \text{ is Fisher separable from } \boldsymbol{x}_i.$$

According to Definition 3, this is equivalent to that $(\boldsymbol{x}, \boldsymbol{x}) - (\boldsymbol{x}, \boldsymbol{x}_i) > 0$. Therefore

$$P(\text{not } A_i) = \int_{(\boldsymbol{x},\boldsymbol{x})-(\boldsymbol{x},\boldsymbol{x}_i)\leqslant 0} p(\boldsymbol{x}) d\boldsymbol{x}.$$

According to the De Morgan's law,

$$\bigwedge_{i=1}^{M} A_i = \text{not}\left(\bigvee_{i=1}^{M} (\text{not } A_i)\right).$$

Hence

$$P(A_1 \wedge A_2 \wedge \cdots \wedge A_M) = 1 - P((\text{not } A_1) \vee (\text{not } A_2) \vee \cdots \vee (\text{not } A_M)),$$

and consequently

$$P(A_1 \wedge A_2 \wedge \cdots \wedge A_M) \geqslant 1 - \sum_{i=1}^{M} P(\text{not } A_i). \tag{6}$$

Consider the set

$$\Omega_i = \{\boldsymbol{x} \in \mathbb{B}_n | (\boldsymbol{x}, \boldsymbol{x}) - (\boldsymbol{x}, \boldsymbol{x}_i) \leqslant 0\} = \mathbb{B}_n(\|\boldsymbol{x}_i\|/2, \boldsymbol{x}_i/2).$$

Since $\boldsymbol{x}_i \in \mathbb{B}_n$, it follows that $\Omega_i \subseteq \mathbb{B}_n(1/2, \boldsymbol{x}_i/2)$. This and Assumption 1 imply that

$$\int_{(\boldsymbol{x},\boldsymbol{x})-(\boldsymbol{x},\boldsymbol{x}_i)\leqslant 0} p(\boldsymbol{x}) d\boldsymbol{x} \leqslant \int_{\mathbb{B}_n(1/2,\boldsymbol{x}_i/2)} p(\boldsymbol{x}) d\boldsymbol{x} \leqslant C\left(\frac{r}{2}\right)^n \tag{7}$$

Combining (7) and (6) we can conclude that the probability that $\boldsymbol{x}$ is separable from all $\boldsymbol{x}_i$ is bounded from below by the expression in (5). □

Remark 1. According to Theorem 1, a single-point set $\mathcal{Y}$ under some mild hypotheses can be separated from $\mathcal{X}$ by a simple Fisher discriminant with probability close to one. Denoting

$$\delta = MC\left(\frac{r}{2}\right)^n$$

one can conclude that $\mathcal{Y}$ is Fisher separable from a given set $\mathcal{X} \subset \mathbb{B}_n$ with probability grater or equal to $1 - \delta$ if

$$n \geqslant \frac{\log M + \log C - \log \delta}{\log 2 - \log r}.$$

For $M = 10^4, C = 10^2, \delta = 10^{-3}$, and $r = 1$, the above inequality holds for all $n \geqslant 30$.

In practice, however, we may be interested in a situation when the set $\mathcal{Y}$ contains several elements which may or may not have some inherent clustering structure or concentrations. In what follows we derive a generalization of Theorem 1 enabling us to formally address the latter case too.

Consider two random sets $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ and $\mathcal{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K\}$. Let there be a process (e.g. a learning algorithm) which, for the given $\mathcal{X}, \mathcal{Y}$ or their subsets, produces a classifier

$$f(\cdot) = \sum_{i=1}^{d} \alpha_i(\boldsymbol{z}_i, \cdot), \ \alpha_i \in \mathbb{R}.$$

The vectors $\boldsymbol{z}_i, i = 1, \ldots, d$ are supposed to be known. Furthermore, we suppose that the function f is such that

$$f(\boldsymbol{y}_j) > \sum_{m,k=1}^{d} \alpha_m \alpha_k (\boldsymbol{z}_m, \boldsymbol{z}_k) \tag{8}$$

for all $\boldsymbol{y}_j \in \mathcal{Y}$. In other words, if we denote $\boldsymbol{w} = \sum_{i=1}^{d} \alpha_i \boldsymbol{z}_i$, the following holds true:

$$(\boldsymbol{w}, \boldsymbol{w}) < (\boldsymbol{w}, \boldsymbol{y}_i) \text{ for all } i = 1, \ldots, K. \tag{9}$$

Note that since the $\mathcal{Y}, \mathcal{X}$ are random, it is natural to expect that the vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$ is also random. The following statement can now be formulated:

Theorem 2. *Consider sets $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ and $\mathcal{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K\}$. Let $p_\alpha(\boldsymbol{\alpha})$ be the probability density function associated with the random vector $\boldsymbol{\alpha}$, and $\boldsymbol{\alpha}$ satisfies condition (8) with probability 1. Then the set $\mathcal{X}$ is separable from the set $\mathcal{Y}$ with probability*

$$P \geqslant 1 - \sum_{i=1}^{M} \int_{H(\boldsymbol{\alpha}, \boldsymbol{x}_i) \leqslant 0} p_\alpha(\boldsymbol{\alpha}) d\boldsymbol{\alpha}, \tag{10}$$

where

$$H(\boldsymbol{\alpha}, \boldsymbol{x}_i) = \sum_{k,m=1}^{d} \alpha_k \alpha_m (\boldsymbol{z}_k, \boldsymbol{z}_m) - \sum_{m=1}^{d} \alpha_m (\boldsymbol{z}_m, \boldsymbol{x}_i).$$

Proof of Theorem 2. Consider events

$$A_i : (\boldsymbol{w}, \boldsymbol{w}) > (\boldsymbol{w}, \boldsymbol{x}_i).$$

Event A_i is equivalent to that the inequality

$$H(\boldsymbol{\alpha}, \boldsymbol{x}_i) = \left(\sum_{k=1}^{d} \alpha_k \boldsymbol{z}_k, \sum_{k=m}^{d} \alpha_m \boldsymbol{z}_m \right) - \sum_{m=1}^{d} \alpha_m (\boldsymbol{z}_m, \boldsymbol{x}_i) > 0$$

holds true. According to (6), Eq. (10) is a lower bound for the probability that all these events hold. Recall that vectors $\boldsymbol{\alpha}$ satisfy (9), and hence

$$\sum_{m=1}^{d} \alpha_m (\boldsymbol{z}_m, \boldsymbol{x}_i) = (\boldsymbol{w}, \boldsymbol{x}_i) < (\boldsymbol{w}, \boldsymbol{y}_j) = \sum_{m=1}^{d} \alpha_m (\boldsymbol{z}_m, \boldsymbol{y}_j)$$

for all $\boldsymbol{x}_i \in \mathcal{X}$ and $\boldsymbol{y}_j \in \mathcal{Y}$ with probability at least (10). The statement now follows immediately from Definition 2. □

Remark 2. Theorem 2 generalizes earlier k-tuple separation theorems [48] to a very general class of practically relevant distributions. No independence assumptions are imposed on the components of vectors $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$. We do, however, require that some information about distribution of the classifier parameters, $\boldsymbol{\alpha}$, is available.

Observe, for example, that if there exist $L > 0, \lambda \in (0, 1)$ and a function $\beta : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ such that

$$\int_{H(\boldsymbol{\alpha}, \boldsymbol{y}) \leqslant 0} p_\alpha(\boldsymbol{\alpha}) d\boldsymbol{\alpha} \leqslant L\lambda^{\beta(d,n)}$$

for any $\boldsymbol{y} \in \mathbb{R}^n$ then (10) becomes

$$P \geqslant 1 - ML\lambda^{\beta(d,n)}.$$

If $d = n$ and elements of the set $\mathcal{Y}$ are sufficiently strongly correlated, then the above bound becomes similar to (5) from Theorem 1. The latter provides a good approximation of the separability probability bound for a simple separating function in which $\boldsymbol{w}$ is just a scaled centroid of the set $\mathcal{Y}$.

Theorems 1 and 2 link dimensionality of the decision-making space with opportunities for quick and fast learning by mere Fisher discriminants. Before, however, moving on to the actual algorithms for either improving legacy data-driven AI systems or generating new edge-based classifiers or both, let us examine another useful property of data in high dimension. This property is summarised in Theorem 3.

Theorem 3. *Let $\boldsymbol{y}$ be a given element of $\mathbb{B}_n$, and let $\mathcal{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_M\}$ be a finite sample of elements $\boldsymbol{x}_i$ drawn identically and independently from a distribution satisfying Assumption 2.*

Then

(1) *$\boldsymbol{y}$ is Fisher separable from the set $\mathcal{X}$ with probability*

$$P \geqslant 1 - \frac{1}{2} MC\left((1 - \|\boldsymbol{y}\|^2)^{\frac{1}{2}} r\right)^n$$

(2) *every $\boldsymbol{x} \in \mathcal{X}$ is Fisher separable from $\boldsymbol{y}$ with probability*

$$P \geqslant 1 - MC(\|\boldsymbol{y}\| r)^n.$$

Proof of Theorem 3. Consider first statement 1) of the theorem. According to Definition 3, the point $\boldsymbol{y}$ is Fisher separable from $\mathcal{X}$ if

$$\|y\|^2 = (\boldsymbol{y}, \boldsymbol{y}) > (\boldsymbol{y}, \boldsymbol{x}) \; \forall \boldsymbol{x} \in \mathcal{X}$$

Given that $\mathcal{X} \in \mathbb{B}_n$, this is equivalent to the fact that no element of $\mathcal{X}$ belongs to the spherical cap

$$\mathcal{C}_n(\boldsymbol{y}) = \left\{ \boldsymbol{z} \in \mathbb{B}_n | \left(\frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}, \boldsymbol{z} \right) - \|\boldsymbol{y}\| \geqslant 0 \right\}.$$

The probability that an $\boldsymbol{x} \in \mathcal{X}$ ends up in the cap is

$$\int_{\mathcal{C}_n(\boldsymbol{y})} p(\boldsymbol{x}) d\boldsymbol{x} \leqslant \frac{1}{2} \int_{\mathbb{B}_n(\sqrt{1-\|\boldsymbol{y}\|^2}\boldsymbol{y})} p(\boldsymbol{x}) d\boldsymbol{x} \leqslant \frac{1}{2} \left((1 - \|\boldsymbol{y}\|^2)^{\frac{1}{2}} r \right)^n.$$

Combining this with (6) gives the required bound.

Statement (2) follows from the observation that all $\boldsymbol{x} \in \mathcal{X}$ outside of the ball $\mathbb{B}_n(\|\boldsymbol{y}\|, 0)$ are Fisher separable from $\boldsymbol{y}$:

$$\|\boldsymbol{x}\| > \|\boldsymbol{y}\| \Rightarrow (\boldsymbol{x}, \boldsymbol{x}) - (\boldsymbol{x}, \boldsymbol{y}) \geqslant \|\boldsymbol{x}\| \|\boldsymbol{y}\| - (\boldsymbol{x}, \boldsymbol{y}) > 0.$$

□

Note that the choice of $\boldsymbol{y}$ in the statement of Theorem 3 is assumed to be independent on the draw of the sample $\mathcal{X}$. Theorem 3 enables us to formulate a simple corollary revealing an interesting dichotomy of datasets in high-dimensional spaces. More precisely.

Corollary 1. Let $\boldsymbol{y}$ be a given element of $\mathbb{B}_n$, and let $\mathcal{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_M\}$ be a finite sample of elements $\boldsymbol{x}_i$ drawn identically and independently from a distribution satisfying Assumption 2 with $r < \sqrt{2}$. Then, with probability

$$P \geqslant 1 - MC\left(\frac{r}{\sqrt{2}}\right)^n,$$

either

(1) $\boldsymbol{y}$ is Fisher separable from $\mathcal{X}$or

(2) every $\boldsymbol{x} \in \mathcal{X}$ is Fisher separable from $\boldsymbol{y}$ and $\boldsymbol{y}$ is inside the ball $\mathbb{B}_n(1/\sqrt{2}, 0)$.

Proof of Corollary 1. Let $\boldsymbol{y}$ be an arbitrary element of $\mathbb{B}_n$. Then either $1 - \|\boldsymbol{y}\|^2 \leqslant \|\boldsymbol{y}\|^2$ or $1 - \|\boldsymbol{y}\|^2 > \|\boldsymbol{y}\|^2$. If the first alternative holds true then $1 - \|\boldsymbol{y}\|^2 \leqslant 1/2$ and statement 1) follows from Theorem 3, alternative 1. If the opposite holds true then statement 2) follows from Theorem 3, alternative 2. □.

Corollary 1 captures and formalizes the following geometric dichotomy of high-dimensional data:

For sufficiently large dimension n, a given point $\boldsymbol{y}$ *in* $\mathbb{B}_n$*, and a finite sample* $\mathcal{X}$ *drawn from a class of distributions (Assumption 2 with* $r < \sqrt{2}$ *and suitable C), with probability close to 1, the point* $\boldsymbol{y}$ *is either Fisher-separable from the sample* $\mathcal{X}$ *or every element* $\boldsymbol{x}$ *of* $\mathcal{X}$ *is Fisher-separable from* $\boldsymbol{y}$*.*

This an other properties Theorems 1–3 reveal important geometrical structures in a broad class of finite high-dimensional data sets. We will exploit these properties in the next section (Section 4) and discuss relationships between these properties and some open questions and observations in the theory and practice of statistical learning (Section 5). We would also like to note that the independence requirements in Theorem 3 and Corollary 1 can be relaxed in various ways, for example, by replacing probability densities in Assumptions 1,2 with appropriate conditional densities (see [12], [17] for more details and ways to relax the independence requirements). With the latter modifications of assumptions, relevant statements and conclusions will not change.

4. Fast removal of AI errors and learning from new examples without catastrophic forgetting

Consider two finite sets, the set $\mathcal{X} \subset \mathbb{R}^n$ representing correct responses of the asset AI system, and the set $\mathcal{Y} \subset \mathbb{R}^n$ representing errors or new knowledge to be accommodated. The task is to efficiently construct a classifier separating the set $\mathcal{X}$ from $\mathcal{Y}$.

According to theoretical constructions presented in the previous section, the following is an advantage for successful and efficient separation of random sets in high dimension: one of the sets (set $\mathcal{Y}$) should be sufficiently concentrated (spatially localized and have an exponentially smaller volume relative to the other [Theorems 1, 2]). If this is the case then, successful separability of this set of smaller volume depends on absence of unexpected concentrations in the probability distributions. Importantly, the probability of success approaches one exponentially fast, as a function of the data dimensionality.

In practice, however, the assumption that one of the sets is spatially localized in a small volume is too restrictive. To overcome this issue, we propose to partition/cluster the set $\mathcal{Y}$ into a union of spatially localized subsets. Presence of local concentrations and separability issues have been linked and analyzed in [15,12,1]. The proposed clustering of the set $\mathcal{Y}$ aims at addressing these issues too.

Below we present an algorithm for fast and efficient error correction of AI systems which is motivated by these observations and intuition stemming from our theoretical results.

Algorithm 1. (Few-shot AI corrector: 1NN version. Training). Input: sets $\mathcal{X}, \mathcal{Y}$, the number of clusters, k, threshold, θ (or thresholds $\theta_1, \dots, \theta_k$).

1. Determining the centroid $\bar{\boldsymbol{x}}$ of the $\mathcal{X}$. Generate two sets, $\mathcal{X}_c$, the centralized set $\mathcal{X}$, and $\mathcal{Y}^*$, the set obtained from $\mathcal{Y}$ by subtracting $\bar{\boldsymbol{x}}$ from each of its elements.
2. Construct Principal Components for the centralized set $\mathcal{X}_c$.
3. Using Kaiser, broken stick, conditioning rule, or otherwise, select $m \leqslant n$ Principal Components, $h_1, \dots, h_m$, corresponding to the first largest eivenvalues $\lambda_1 \geqslant \dots \geqslant \lambda_m > 0$ of the covariance matrix of the set $\mathcal{X}_c$, and project the centralized set $\mathcal{X}_c$ as well as $\mathcal{Y}^*$ onto these vectors. The operation returns sets $\mathcal{X}_r$ and $\mathcal{Y}^*_r$, respectively:

$$\mathcal{X}_r = \{\boldsymbol{x} | \boldsymbol{x} = H\boldsymbol{z}, \boldsymbol{z} \in \mathcal{X}_c\}$$
$$\mathcal{Y}^*_r = \{\boldsymbol{y} | \boldsymbol{y} = H\boldsymbol{z}, \boldsymbol{z} \in \mathcal{Y}^*\}, H = \begin{pmatrix} h_1^T \\ \vdots \\ h_m^T \end{pmatrix}.$$

4. Construct matrix W

$$W = \operatorname{diag}\left(\frac{1}{\sqrt{\lambda_1}}, \dots, \frac{1}{\sqrt{\lambda_m}}\right)$$

corresponding to the whitening transformation for the set $\mathcal{X}_r$. Apply the whitening transformation to sets $\mathcal{X}_r$ and $\mathcal{Y}^*_r$. This returns sets $\mathcal{X}_w$ and $\mathcal{Y}^*_w$:

$$\mathcal{X}_w = \{\boldsymbol{x} | \boldsymbol{x} = W\boldsymbol{z}, \boldsymbol{z} \in \mathcal{X}_r\}$$
$$\mathcal{Y}^*_w = \{\boldsymbol{y} | \boldsymbol{y} = W\boldsymbol{z}, \boldsymbol{z} \in \mathcal{Y}^*_r\}.$$

5. Cluster the set $\mathcal{Y}^*_w$ into k clusters $\mathcal{Y}^*_{w,1}, \dots, \mathcal{Y}^*_{w,k}$. Let $\bar{\boldsymbol{y}}_1, \dots, \bar{\boldsymbol{y}}_k$ be their corresponding centroids.
6. For each pair $(\mathcal{X}_w, \mathcal{Y}^*_{w,i}), i = 1, \dots, k$, construct (normalized) Fisher discriminants $\boldsymbol{w}_1, \dots, \boldsymbol{w}_k$:

$$\boldsymbol{w}_i = \frac{(\operatorname{Cov}(\mathcal{X}_w) + \operatorname{Cov}(\mathcal{Y}^*_{w,i}))^{-1}\bar{\boldsymbol{y}}_i}{\|(\operatorname{Cov}(\mathcal{X}_w) + \operatorname{Cov}(\mathcal{Y}^*_{w,i}))^{-1}\bar{\boldsymbol{y}}_i\|}.$$

An element $\boldsymbol{z}$ is associated with the set $\mathcal{Y}^*_{w,i}$ if $(\boldsymbol{w}_i, \boldsymbol{z}) > \theta$ and with the set $\mathcal{X}_w$ if $(\boldsymbol{w}_i, \boldsymbol{z}) \leqslant \theta$.

If multiple thresholds are given then an element $\boldsymbol{z}$ is associated with the set $\mathcal{Y}^*_{w,i}$ if $(\boldsymbol{w}_i, \boldsymbol{z}) > \theta_i$ and with the set $\mathcal{X}_w$ if $(\boldsymbol{w}_i, \boldsymbol{z}) \leqslant \theta_i$.

Output: vectors $\boldsymbol{w}_i, i = 1, \ldots, k$, matrices H and W.

Remark 3. It is worthwhile to note that Steps 3,4 of Algorithm 1 can in principle be extended or supplemented with exhaustive automated or semi-automated subspace learning [33] with an aim to produce best features to build AI corrections on. These extensions, however, may be dependent on human input or they may be incompatible with computational resources available. A plausible compromise could be to find suitable subspaces offline and replace Steps 3,4 with computing projections of the data onto these subspaces followed by data whitening.

The deployment/application part of the algorithm is as follows:

Algorithm 2. (Few-shot AI corrector: 1NN version. Deployment). Input: a data vector $\boldsymbol{x}$, the set's $\mathcal{X}$ centroid vector $\bar{\boldsymbol{x}}$, matrices H, W, the number of clusters, k, cluster centroids $\bar{\boldsymbol{y}}_1, \ldots, \bar{\boldsymbol{y}}_k$, threshold, θ (or thresholds $\theta_1, \ldots, \theta_k$), discriminant vectors, $\boldsymbol{w}_i, i = 1, \ldots, k$.

1. Compute
$$\boldsymbol{x}_w = WH(\boldsymbol{x} - \bar{\boldsymbol{x}})$$
2. Determine
$$\ell = \arg\min_i \|\boldsymbol{x}_w - \bar{\boldsymbol{y}}_i\|.$$
3. Associate the vector $\boldsymbol{x}$ with the set $\mathcal{Y}$ if $(\boldsymbol{w}_\ell, \boldsymbol{x}_w) > \theta$ and with the set $\mathcal{X}$ otherwise.
If multiple thresholds are given then associate the vector $\boldsymbol{x}$ with the set $\mathcal{Y}$ if $(\boldsymbol{w}_\ell, \boldsymbol{x}_w) > \theta_\ell$ and with the set $\mathcal{X}$ otherwise.

Output: a label attributed to the vector $\boldsymbol{x}$.

In contrast to previously proposed approaches using stochastic separation effects [48], Algorithm 2 mitigates the presence of clusters whose centroids are close to the origin. If such clusters do occur and the fraction of the set $\mathcal{X}$ located in their vicinity is not overwhelmingly large (which is ensured by Theorem 3 and Corollary 1) then, at the stage of deployment, the corresponding correcting discriminants will be triggered by elements from $\mathcal{X}$ infrequently.

Remark 4. According to Theorem 1, a single-point set $\mathcal{Y}$ under some mild hypotheses would be separated from $\mathcal{X}$ with probability close to one.

If the set $\mathcal{Y}$ consists of multiple correlated subsets then, as the number of clusters increases, one would expect that the algorithm's performance in separating the sets $\mathcal{X}, \mathcal{Y}$ improves.

At the same time, one may not necessarily require a near-perfect separability. For example, removal of 90% of all errors at the cost of a slight performance degradation of the AI's basic functionality may be an acceptable compromise in many applications. If the data dimensionality is sufficiently high then the desired separation might be achieved with just a single linear functional, provided that the centroid $\bar{\boldsymbol{y}}$ of the set $\mathcal{Y}$ is separated away from the centroid $\bar{\boldsymbol{x}}$ of the set $\mathcal{X}$.

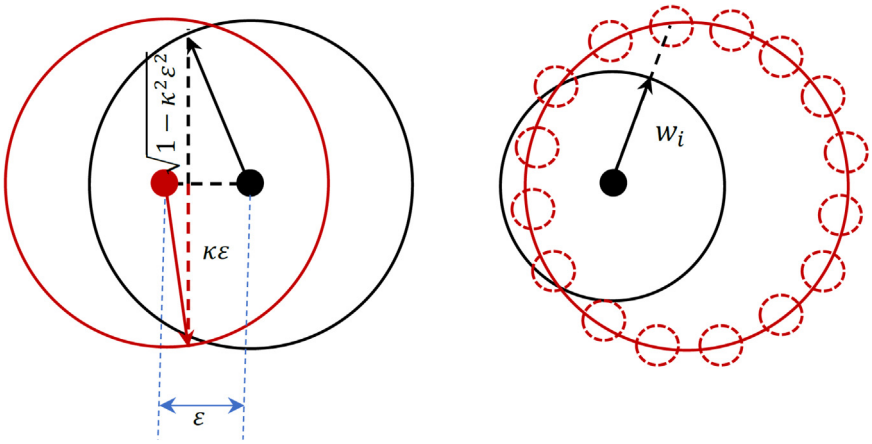

Fig. 1. Separation of a non-isolated set $\mathcal{Y}$ from $\mathcal{X}$. Black circle represents the set $\mathcal{X}$, red circle represents the set $\mathcal{Y}$. Filled disks represent centres of the sets $\mathcal{X}$ and $\mathcal{Y}$, respectively. *Left panel.* Single-cluster case. *Right panel.* Multiple-cluster case. According to Theorem 3, the classes are "hollow" inside if Assumption 2 with appropriate parameters holds for their correspondin.g distributions.

The rationale behind this observation is as follows. Let $\mathcal{X}$ be equidistributed in $\mathbb{B}_n$ and $\mathcal{Y}$ be drawn from another equidistribution in a unit n-ball but centered at a point whose Euclidean norm is $0 < \varepsilon \ll 1$ (see Fig. 1). Let $\boldsymbol{w} = (\bar{\boldsymbol{y}} - \bar{\boldsymbol{x}})/\|\bar{\boldsymbol{y}} - \bar{\boldsymbol{x}}\| = \varepsilon^{-1}\bar{\boldsymbol{y}}$. Let $\kappa \in (0,1)$, and let $h(\boldsymbol{x}) = (\boldsymbol{x}, \boldsymbol{w}) - \kappa\varepsilon$ be the separating hyperplane so that if $h(\boldsymbol{z}) > 0$ then the vector $\boldsymbol{z}$ is associated with $\mathcal{Y}$, and $\boldsymbol{z}$ is associated with the set $\mathcal{X}$ if $h(\boldsymbol{z}) \leqslant 0$. Then the fraction of elements from $\mathcal{X}$ "missed" (false negative response) by this rule is bounded from above by

$$\rho_x = \frac{1}{2}(1 - \kappa^2\varepsilon^2)^{\frac{n}{2}},$$

and the fraction of elements from $\mathcal{Y}$ incorrectly attributed to $\mathcal{X}$ (false positive response) is bounded from above by

$$\rho_y = \frac{1}{2}\left(1 - (1-\kappa)^2\varepsilon^2\right)^{\frac{n}{2}}.$$

Hence, when n is sufficiently large, both ρ_x, ρ_y may be made acceptably small even if ε is small too (cf [16]).

Remark 5. Note that if the clustering step in Algorithm 1 is performed so that, for every i

$$\boldsymbol{z} \in \mathcal{Y}^*_{w,i} \Rightarrow \|\boldsymbol{z} - \bar{\boldsymbol{y}}_i\| < \|\boldsymbol{z} - \bar{\boldsymbol{y}}_j\|, \text{ for all } j \neq i$$

then the proposed 1NN integration logic, as in Algorithm 2, correctly assigns elements from $\mathcal{Y}$ to their corresponding discriminants $\boldsymbol{w}_i$. For each query point, only one of the linear discriminants is active. This is markedly different from the more aggressive "union" integration logic (OR rule) proposed in [47] in which, for a single query point, all discriminants are active simultaneously leading to higher chances of producing false negative errors. In this respect, the 1NN rule is somewhat tighter than the OR rule.

Remark 6. It may sometimes be computationally advantageous to perform the clustering step in Algorithm 1 (step 5) prior to dimensionality reduction. This will result in that the deployment part of the algorithm, Algorithm 2 changes as follows

Algorithm 3.

(Few-shot AI corrector: 1NN version. Deployment) Input: a data vector $\boldsymbol{x}$, the set's $\mathcal{X}$ centroid vector $\bar{\boldsymbol{x}}$, matrices H, W, the number of clusters, k, cluster centroids $\bar{\boldsymbol{y}}_1, \ldots, \bar{\boldsymbol{y}}_k$, threshold, θ (or thresholds $\theta_1, \ldots, \theta_k$), discriminant vectors, $\boldsymbol{w}_i, i = 1, \ldots, k$.

Pre-compute vectors

$$\boldsymbol{w}^*_i = H^T W \boldsymbol{w}_i.$$

1. Determine
$$\ell = \arg\min_i \|\boldsymbol{x} - \bar{\boldsymbol{x}} - \bar{\boldsymbol{y}}_i\|.$$
2. Associate the vector $\boldsymbol{x}$ with the set $\mathcal{Y}$ if $(\boldsymbol{w}^*_\ell, (\boldsymbol{x} - \bar{\boldsymbol{x}})) > \theta$ and with the set $\mathcal{X}$ otherwise.
If multiple thresholds are given then associate the vector $\boldsymbol{x}$ with the set $\mathcal{Y}$ if $(\boldsymbol{w}^*_\ell, (\boldsymbol{x} - \bar{\boldsymbol{x}})) > \theta_\ell$ and with the set $\mathcal{X}$ otherwise.

The difference between Algorithm 2 and 3 is that the data point $\boldsymbol{x}$ no longer needs to be projected onto the principal components. This may be computationally advantageous when the number of components on which the data is projected is larger than the number of clusters k.

5. Discussion

Results and algorithms presented in Sections 3.2, 4 enable equipping existing asset AI systems with capabilities to learn from new examples via transparent and reversible modifications of their structure in response to errors. Modifications of the systems' structure are:

- nodes and components implementing calculations of relevant inner products determined in Steps 2 and 3 in Algorithms 3 and 2, respectively
- relevant decision-making/ integration logic in these steps.

In what follows we discuss other relevant properties of the proposed algorithms such as performance bounds, their relation to existing approaches in the filed of few-shot learning, and possible future directions.

5.1. Guaranteed performance bounds

One of the main features of our approach is that it enables explicit estimation and control of potential negative performance changes induced by learning/ error correction (Algorithms 1–3).

Indeed, let Algorithms 1, 3 return weights $\boldsymbol{w}_1^*, \ldots, \boldsymbol{w}_k^* \in \mathbb{R}^n$, thresholds $\theta_1, \ldots, \theta_k$, and the value of $\bar{\boldsymbol{x}}$. Let $\boldsymbol{z}_i - \bar{\boldsymbol{x}}, i = 1, \ldots, N$ be a finite i.i.d. sample drawn from a distribution defined on $\mathbb{B}_n$ and satisfying Assumption 2. Suppose now that this sample represents new data which Algorithms 1, 3 did not have access to and on which performance of the asset AI system prior to the application of Algorithms 1, 3 was correct. In the context of Algorithms 1 and 3, vectors $\boldsymbol{z}_i$ are to be associated with the set $\mathcal{X}$. According to Algorithm 3, an element $\boldsymbol{z}_i$ is assessed by a single ℓ-th discriminant. If

$$(\boldsymbol{w}_\ell^*, \boldsymbol{z}_i - \bar{\boldsymbol{x}}) \leqslant \theta_\ell$$

then the modified asset AI correctly associates this element with the set $\mathcal{X}$. If, however,

$$(\boldsymbol{w}_\ell^*, \boldsymbol{z}_i - \bar{\boldsymbol{x}}) > \theta_\ell$$

then the modified system assigns $\boldsymbol{z}_i$ to $\mathcal{Y}$. This assignment, if the set $\mathcal{Y}$ corresponds to errors, introduces an error in the combined system.

The probability P_e of such an error can be bounded from above as

$$P_e \leqslant \frac{1}{2} C \left(\left(1 - \left(\frac{\theta_\ell}{\|\omega_\ell^*\|} \right)^2 \right)^{\frac{1}{2}} r \right)^n. \tag{11}$$

If there is an $v \in (0, 1)$ such that

$$C^{\frac{1}{n}} \left(1 - \left(\frac{\theta_\ell}{\|\omega_\ell^*\|} \right)^2 \right)^{\frac{1}{2}} r \leqslant v \tag{12}$$

for all $\ell = 1, \ldots, k$ then $P_e \leqslant \frac{1}{2} v^n$ converges to 0 exponentially fast with n. This probability *does not grow with k as long as (12) holds true*. This is an advantage over earlier versions of AI corrector [47,13,48] in which worst-case bounds on P_e grow linearly with k as a consequence of the conjunction (OR) integration logic.

The proposed few-shot AI correction mechanisms complement existing approaches to learning without catastrophic forgetting such as elastic weight consolidation [27]. Indeed, under appropriate conditions, it follows from (11), (12) that new knowledge acquired through the application of Algorithms 1, 3 has an exponentially vanishing probability to damage to AI existing skills. For given and fixed bounds on parameters C, r of the data distribution (Assumption 2), this probability can be controlled by the ratio $\frac{\theta_\ell}{\|\omega_\ell^*\|}$.

5.2. Geometry of few- or one-shot learning

Theorems 1–3, on which our algorithms are built, enable to shed additional light on why and when few- and one-shot learning works [51,53] (see also references therein). Consider task-invariant embedding learning [51] for a binary classification task. Let $\{0, 1\}$ be the set of labels. Following [51,53], let the predicted label $\hat{p}(\boldsymbol{z})$ of the vector $\boldsymbol{z}$ be defined as

$$\hat{p}(\boldsymbol{z}) = F\left(\sum_{i=1}^m a(\boldsymbol{y}_i, \boldsymbol{z}) p_i(\boldsymbol{y}_i) - \theta \right), \tag{13}$$

where $a : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a kernel, $F : \mathbb{R} \to \mathbb{R}$ is a label matching function, $\boldsymbol{y}_i \in \mathbb{R}^n$ are examples to learn from, $p_i(\boldsymbol{y}_i)$ are weights, and F is a step function:

$$F(s) = \begin{cases} 1, & s > 0 \\ 0, & s \leqslant 0. \end{cases}$$

The kernel function $a(\cdot, \cdot)$ "matches" data to existing knowledge, and the function F assigns labels to data on the basis of this matching.

In our settings, kernel function $a(\cdot, \cdot)$ is an inner product $(\cdot, \cdot)$. Let $p_i(\boldsymbol{y}_i) = 1/m$. Hence (13) becomes

$$\hat{p}(\boldsymbol{z}) = F((\boldsymbol{w}, \boldsymbol{z}) - \theta), \boldsymbol{w} = \frac{1}{m} \sum_{i=1}^m \boldsymbol{y}_i, \tag{14}$$

where $\boldsymbol{w}$ is a *class prototype* [53,42] (cf [21]). Let υ, Δ, δ be numbers such that

$$\|\boldsymbol{y}_i\|^2 > \upsilon \text{ for all } i \in \{1, \ldots, m\} \tag{15}$$

and

$$\Delta \geqslant (\boldsymbol{y}_i, \boldsymbol{y}_j) \geqslant \delta > 0 \; \forall i \neq j, \; i,j \in \{1,\dots,m\}. \tag{16}$$

The latter condition can be interpreted as a consistency requirement in the sense that all training samples in the same learning episode share a degree of similarity.

Set

$$\theta = \frac{1}{m}\upsilon + \frac{m-1}{m}\delta. \tag{17}$$

It now follows that $(\boldsymbol{w},\boldsymbol{y}) > \theta$ for every vector from the convex hull of $(\boldsymbol{y}_1,\dots,\boldsymbol{y}_m)$:

$$(\boldsymbol{w},\boldsymbol{y}) > \theta \text{ for all } \boldsymbol{y} \in \mathrm{Conv}(\boldsymbol{y}_1,\dots,\boldsymbol{y}_m).$$

Indeed, if $\boldsymbol{y} = \sum_{i=1}^m \lambda_i \boldsymbol{y}_i$, $\lambda_i \geqslant 0, \sum_{i=1}^m \lambda_i = 1$ then

$$(\boldsymbol{w},\boldsymbol{y}) = \sum_{i=1}^m \lambda_i (\boldsymbol{w},\boldsymbol{y}_i) > \sum_{i=1}^m \lambda_i \left(\frac{1}{m}\upsilon + \frac{m-1}{m}\delta\right) = \left(\frac{1}{m}\upsilon + \frac{m-1}{m}\delta\right).$$

Geometrically, as is shown in Fig. 2, $\mathrm{Conv}(\boldsymbol{y}_1,\dots,\boldsymbol{y}_m)$ and its appropriate ε-thickening can be viewed as a "knowledge core" or a "knowledge slab" of the training vectors $\boldsymbol{y}_1,\dots,\boldsymbol{y}_m$.

Class prototype vector $\boldsymbol{w}$ induces a decision rule labeling all points $\boldsymbol{y}:\ (\boldsymbol{w},\boldsymbol{y}) > \theta$ as "1". Using (11), were $\boldsymbol{w}^*_\ell, \theta$ are replaced with $\boldsymbol{w}, \theta$ from (14), (17), one can determine probability bounds on how acquiring new knowledge through few-shot schemes like (13) affects assets AI skills on other tasks.

Note that few-shot rules (13), (14), in addition to learning "knowledge core", may associate many more points with the label "1". These extra points form "knowledge shadow" as is shown in Fig. 2. The volume of the "knowledge shadow" may exceed that of the "knowledge core".

Let $R = \left(1 - (\theta/\|\boldsymbol{w}\|)^2\right)^{1/2}$ denote the radius of the disc in the base of the spherical cap: $C(\boldsymbol{w},\theta) = \{\boldsymbol{y} \in \mathbb{B}_n | (\boldsymbol{w},\boldsymbol{y}) > \theta\}$. The set

$$\mathrm{Sh} = C(\boldsymbol{w},\theta) \setminus \mathrm{Conv}(\boldsymbol{y}_1,\dots,\boldsymbol{y}_m)$$

is the "knowledge shadow". It is well-known that the volume of the convex hull of any m vectors in $\mathbb{B}_n(R,0)$ is bounded from above by $m/2^n V(\mathbb{B}_n(R,0))$ [7]. At the same time $V(C(\boldsymbol{w},\theta)) > 1/n(1-(1-R^2))^{1/2} V(\mathbb{B}_{n-1}(R,0))$. Hence

$$\frac{V(\mathrm{Sh})}{V(C(\boldsymbol{w},\theta))} > 1 - \frac{mn}{2^n}\frac{R}{(1-(1-R^2))^{1/2}}\frac{V(\mathbb{B}_n(1,0))}{V(\mathbb{B}_{n-1}(1,0))}.$$

Therefore in high dimension the volume of the "knowledge shadow" becomes exponentially large relative to that of the "knowledge core". Moreover, since the prototype class $\boldsymbol{w}$ is a sample average, the scheme inherits a degree of robustness to perturbations of the training data $\boldsymbol{y}_1,\dots,\boldsymbol{y}_m$. These properties, as well as (15), (16), explain when and why few- and one-shot learning algorithms such as Algorithms 1–3 or [42,51] are robust and generalise offering a solution pathway to the challenge of generalisaion in large-scale systems [54].

A natural extension of few-shot learning schemes (13) is to allow weights $p(\boldsymbol{y}_i)$ to depend on the values of $\boldsymbol{z}$. One of the advantages of such extension is a capability to learn from training data for which consistency condition (13) does not hold true. Indeed, in this case training data can be partitioned into clusters satisfying (13). For every cluster there will be a prototype that is activated by $\boldsymbol{z}$. Proposed Algorithms 1–3 are examples of such extended schemes.

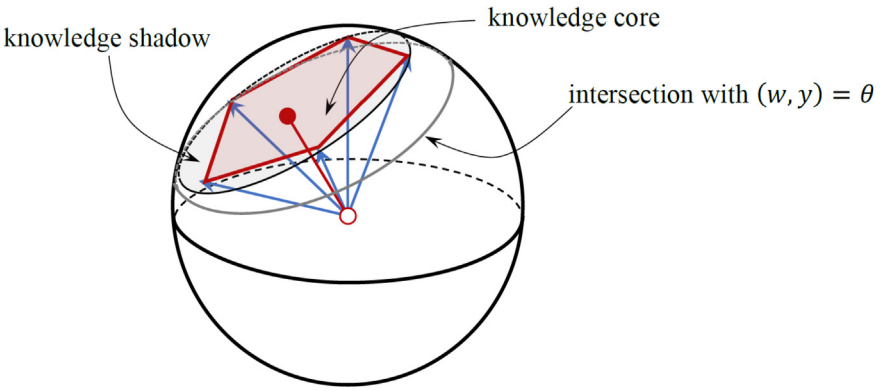

Fig. 2. Geometrical interpretation of few-shot learning. Vertices of the pentagon are training samples $\boldsymbol{y}_1,\dots,\boldsymbol{y}_m$, and vector with the filled circle head shows class prototype $\boldsymbol{w}$. The interior of the pentagon is the "knowledge core", shaded areas adjacent to the sides of the pentagon are in the "knowledge shadow" - the set of all points in the spherical cap $\{\boldsymbol{y} \in \mathbb{B}_n | (\boldsymbol{w},\boldsymbol{y}) > \theta\}$ which are outside of the "knowledge core".

In the next section we illustrate the application of these algorithms in two practical scenarios of edge-based AI deployment.

6. Examples

The choice of examples is motivated primarily by our intention to illustrate the application of the new algorithm with and without clustering. The first example illustrates how one can take advantage of high-dimensional data for constructing a single discriminant. The second example enables us to show advantages of clustering for a problem where the feature space is genuinely high-dimensional and where the number of errors made by edge-based systems necessitates automated and computationally efficient interventions to be deployed at the edge. In both examples, AI correctors were built on the raw input data. This enabled us to emulate extreme scenario where internal signals from the asset AI system are not available. The approach, however, would not change if these signals become available.

6.1. Real-time Tool Wear and Product Quality Prediction for Computer Numerical Controlled (CNC) Milling Machines

6.1.1. System overview and setup

Milling is a process of removing excess material by advancing a cutter into a work piece. It is one of the most commonly used processes for machining freeform surfaces and custom parts [30]. Quality of the final part depends on many factors including tool path, tool orientation, tool geometry, tool wear and security of the part fixing. Here we focused on real-time prediction of part quality from the measurements characterizing electrical and mechanical state of the CNC machine.

To build a data-driven quality detector we used the CNC Milling Dataset from the University of Michigan Smart Lab [45]. The dataset contains a series of machining experiments run on 2” $\times$ 2” $\times$ 1.5” wax blocks. Machining data was collected from a CNC machine for variations of tool condition, feed rate, and clamping pressure. An example of a finished wax part with an “S” shape - S for smart manufacturing - carved into the top face is shown in Fig. 3.

Time series data from the machine was collected from 18 experiments with a sampling rate of 100 ms, and each instantaneous measurement vector contained 48 attributes. The task was to predict the output quality of the part, as confirmed by visual inspection, from the instantaneous measurements.

6.1.2. Automated part quality prediction

In this problem, we used Algorithms 1, 2 with a single cluster. We also used a slightly modified Step 3 in 1 in which we retained 21 Principal Components: from the 20th to the 40th. Note that we did not use the first 19 components as in this particular problem inclusion of these components did not translate into noticeable changes of the model’s performance. Discarding certain principal components is a common practice in many data science domains. For example, first principal components are frequently considered to be associated with technical artifacts in the analysis of omics datasets in bioinformatics, and their removal might improve the downstream analyses [43,4,22]. In some cases in this field, more than 10 first principal components have to be removed, to increase the signal/noise ratio [28].

Training and validation datasets. To train the model we used 9 experiments (out of the total 18) in which 5 experiments corresponded to machining with unworn tool and which passed a visual inspection (experiments $1, 3, 4, 5, 11$) and 4 experiments contained data corresponding to runs that either did not finish or where the part did not pass visual inspection (experiments $6, 7, 8, 9$); 3 experiments were used for testing (experiments 2 and 17 corresponding to successful runs where parts passed visual inspection, and experiment 10 in which the part failed the inspection).

Experiments and results. Each measurement in both training and testing datasets have been labeled as “pass” or “fail” depending on whether the run from which this measurement was taken passed the visual inspection (and hence the point was labeled as a “pass”) or failed (resulting in the label “fail”). Training of the model took 0.16 to 0.2 seconds on a core i7 laptop, and a summary of the model performance is summarized in Fig. 4.

Fig. 3. An example of a part produced by the CNC milling machine .[45].

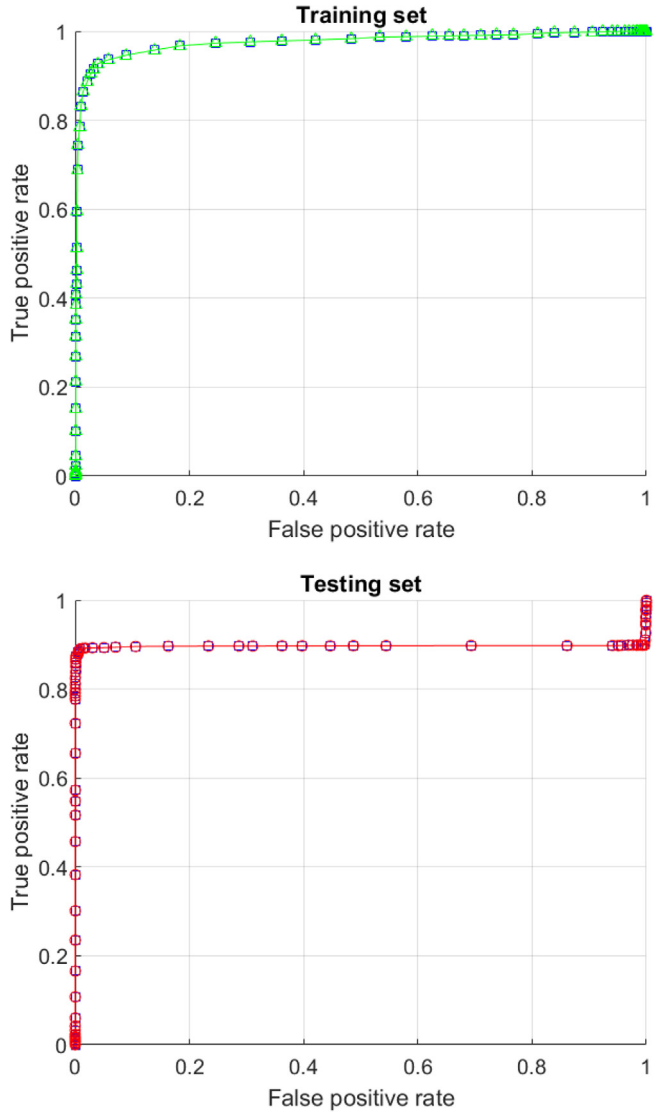

Fig. 4. ROC curves for detecting a faulty run from a single measurement in a run.

To better integrate the model's decision and filter spurious errors we averaged the model's binary output (computed with the threshold of -1.5) over a sliding window of 200 measurements. The resulting value for each window was recorded and shown as "Score" in Fig. 5. Outcomes of these experiments on the entire training and test sets, where the data from several runs was simply concatenated, are presented in Fig. 5.

As one can see from these figures, the model correctly identifies failed runs. Good generalization performance of this simple model can be explained by the concentration effects captured in Fig. 1: relatively small differences of class means are apparently sufficient to ensure reasonable class separation by a Fisher discriminant if the data dimension is sufficiently large.

In addition to this, we note that not every single dimension is equally important. To illustrate this point, let $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{y}}$ denote empirical class means for the "pass" and "fail" classes. We calculated

$$\text{Relative relevance}_i = \left| \frac{\bar{\boldsymbol{x}} - \bar{\boldsymbol{y}}}{\|\bar{\boldsymbol{x}} - \bar{\boldsymbol{y}}\|} \frac{h_i}{\|h_i\|} \right|, \ i = 20, \ldots, 40,$$

where h_i are the corresponding Principal Components (Algorithm 1). The relative relevance indices for each i-th component and the contribution of that component to the total empirical data variance are shown in Fig. 6. As we can see from Fig. 6, a large proportion of principal components only marginally project onto the vector $\bar{\boldsymbol{x}} - \bar{\boldsymbol{y}}$. We would like to comment that high relative relevance values do not necessarily imply that the corresponding features are best for classification in isolation from other features. High relative relevance values, however, may suggest that in the overall high-dimension space, faulty parts

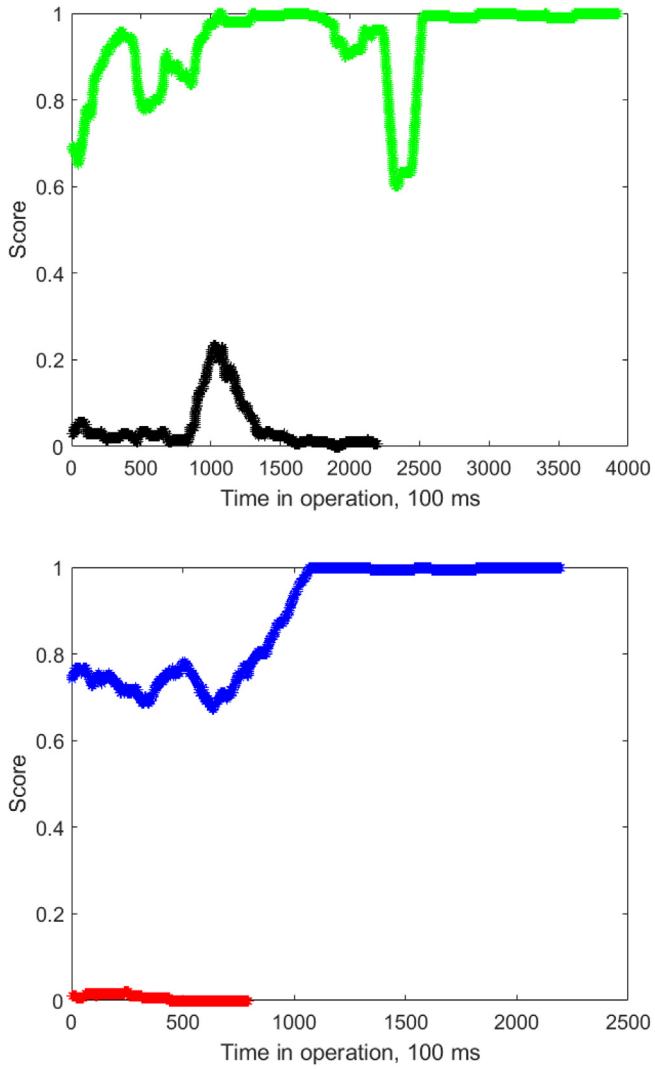

Fig. 5. Averaged prediction "Score" for training (top panel) and testing runs (bottom panel). Green and black curves in the top panel show correct and failed machining for the training set, respectively. Blue and red curves correspond to correct and failed runs in the test set, respectively.

data concentrate near some lower-dimensional subspace. This, in view of discussion in Section 5, could be another factor explaining good practical results of our algorithms in this task.

6.2. Adaptive Removal of False Positives in Video Surveillance and Analytics Systems: A Case Study

6.2.1. System overview and setup

All UK airports are expected to ensure that the average passenger spends no longer than twelve minutes going through the security area.

The current boarding gates can measure the number of passengers coming through the security area via passenger boarding passes. However there is no way of determining whether an individual passenger has left the security area. The current solution involves a member of staff manually keeping track of a small sample of passengers passing from the boarding gates to the security scanners and logging the time taken.

Knowledge about the length of these queues, as well as the number of passengers getting through the airport, helps airports to manage their resources in an efficient way by enabling them to decide how many security stations should be open. It also provides passengers with valuable information on the amount of time they can expect to spend inside the queues thereby allowing them to manage their time inside the airport more efficiently. Thus, knowing the time taken for a passenger since entering boarding-pass gates until leaving security gates, in almost real time, would be very beneficial. However, the current practices aimed at addressing this specific problem are far from being efficient.

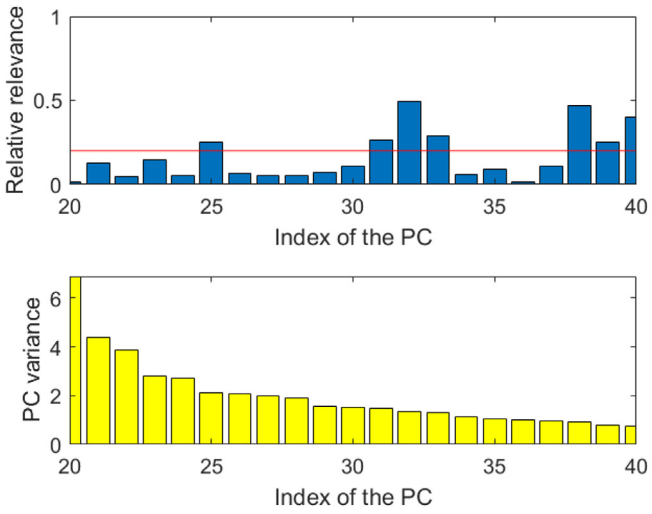

Fig. 6. Relative relevance of Principal Components for class discrimination.

To address the efficiency issue, various Queue Management System (QMS) are being developed. Here we provide a short description of a system which has been developed by the Visual Management Systems Ltd within the scope of the Innovate UK Knowledge Transfer Partnership project (KTP 10522).

The system consists of two major components: the hardware that is in charge of detecting faces, and the back-end server that processes the data streamed from the hardware unit and calculates the average, fastest and slowest security queue times. A front-end web-page is served by the back-end to display the aforementioned statistical data as well as producing historical reports. As mentioned before, security queue time is the time a passenger spends in queues for the Boarding Pass Gates (BPG) and Security Gates (SG).

For the hardware, shown in Fig. 7, we utilise two (or more) high-definition cameras streaming in H.264, one for BPG and one for SG, connected to two *Processing Modules* ^{TM} (PM) via two mini PC's.

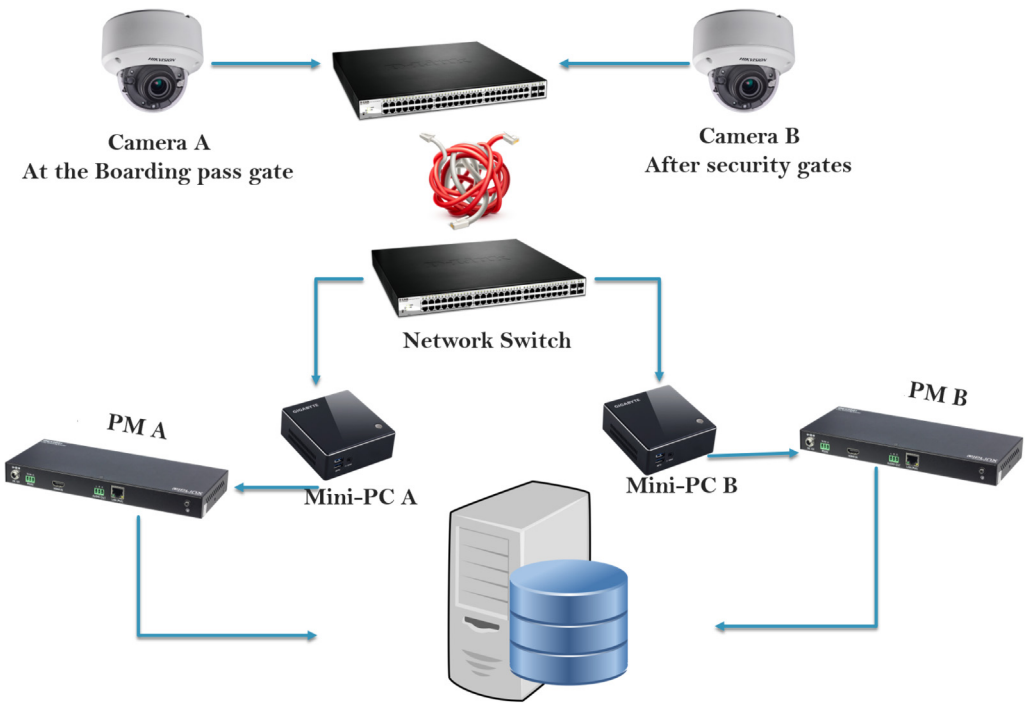

Fig. 7. Queue Management System.

The function of the PMs is to detect and track faces from the video feeds and output relevant metadata (face coordinates, time stamp, bitmap/image of the face etc) to the processing server. The metadata are then received by the back-end server for processing. The back-end includes four different sections; parsing, image pre-processing, feature extraction and the data base. In the parsing section, the encoded face bitmap is extracted together with all other metadata. This is followed by the pre-processing unit in which each thumbnail is encoded to JPG format and saved to the local hard disk before passing through the Dlib library to generate a face score. Then, the thumbnail will pass through a Convolutional Neural Network, e.g. VGG-16 [41], for feature extraction. Finally, features of the thumbnail together with all related metadata will be stored in the database.

The system has been trialed over a period of 3 months in a major UK airport. The trials revealed that, depending on operational conditions, PMs based on propriertrary algorithms, occasionally return false positive detects These false positive detects if left untreated, have a capacity to slow down the entire processing pipeline. As the system scales up, dealing with these false positive detects on the side of the server becomes computationally prohibitive. Moreover, the false positive detects, are camera and place-specific, in general. Thus there is a need and a rationale to address these false positive detects at their source.

6.2.2. Adaptive removal of false positives

In order to address the problem of false positives, we implemented and tested the proposed algorithms in this setup. For the purposes of avoiding issues with reproducibility as well as due to the data protection, in what follows we present a detailed account of this implementation in which the PM was an OpenCV implementation of the Haar face detector. The

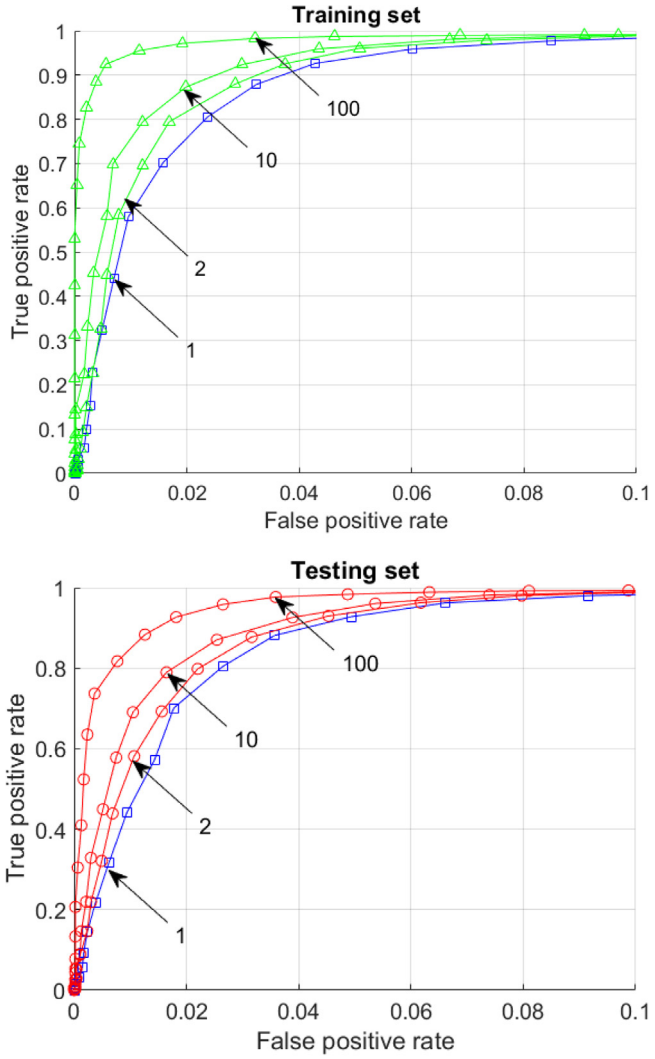

Fig. 8. ROC curves after the application of the AI error correcting algorithm with 200 Principal Components for different numbers of clusters, from 1 to 100.

detector has been applied to a publicly accessible video footage capturing traffic and pedestrians walking on the streets of Montreal. For the purposes of testing and validation, we used standard MTCNN face detector from the mtcnn Python package as a vehicle to generate ground truth data. We did not introduce any changes to parameters of the standard detector from that package. All the data as well as the code generating true positive and false positive images can be found in [35].

For this particular dataset, the total number of true positives was 21896, and the total number of false positives was 9372. All the detects have been resized to 64×64 crops (in RGB encoding). Each crop produces a 12288-dimensional vector. From this dataset, we generated a training set containing 50 percent of positive and false positives, and passed this training set to Algorithm 1. In the algorithm, true positives have been associated with the set $\mathcal{X}$, and false positives were associated with the set $\mathcal{Y}_*$. The number of Principal Components was limited to 200. We did not observe significant performance variations when the number of components changed within 20%. However, as we show later, retaining excessively large number of components may lead to overfit. A possible factor contributing to this effect, in addition to increased Vapnik–Chervonenkis dimension, could be numerical instability of the inversion of matrices in Step 6 of Algorithm 1.

We tried the algorithm for the following numbers of clusters: $1, 5, 10$, and 100. At the deployment stage, we used Algorithm 2. Training took, on average, about 180 s on a Core i7 laptop, and the outcomes of the process as well as performance on the testing set are summarized in Fig. 8 where curves corresponding to different numbers of clusters ($1, 5, 10$, and 100) are annotated by arrows with numbers $1, 5, 10$, and 100, respectively.

As we can see from this figure, even a single-cluster implementation of Algorithm 1 allows one to filter 90 percent of all errors at the cost of missing circa 5 percent of true positives. This is consistent with expectations discussed in Remark 4. Implementation of the single-cluster correcting functional on an ARM Cortex-A53 processor took less than 1 ms per each 12288-dimensional vector implying significant capacity of the approach for embedded and "at the edge" applications.

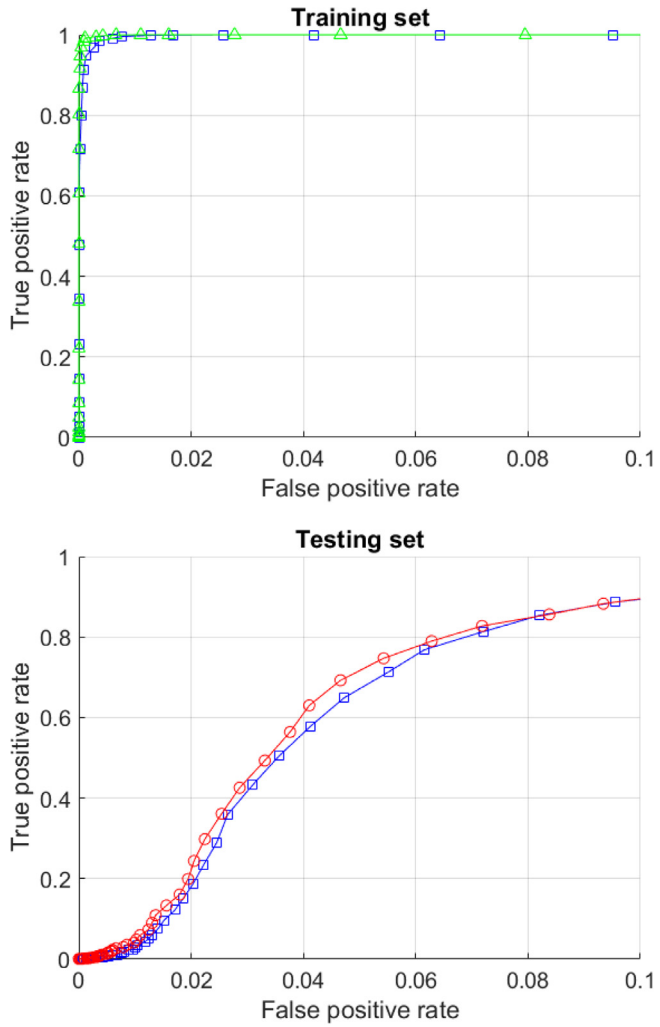

Fig. 9. ROC curves after the application of the AI error correcting algorithm with 6000 Principal Components. Blue curves correspond to the baseline case with a single cluster. Red and green curves show behavior of the system for 100 clusters.

A notable classification performance gain is observed for a 100-cluster version of the algorithm. This, however, comes at additional computational costs at the stage of deployment. Having said this, the deployment part of the algorithm is extremely scalable leading to significant expected reductions of computation times in the case of parallel execution of the code and is hence amenable to massively parallel implementations.

It is also worthwhile to mention that the concentration effects, as formulated in Theorems 1, 2, and which are at the backbone of Algorithms 1–3, may negatively effect the overall system's performance if the dimensionality is excessively high and the cardinality of the set $\mathcal{Y}$ is comparable to that of the set $\mathcal{X}$. To illustrate this point, we used Algorithms 1, 2 with the 6000 Principal Components. Results are shown in Fig. 9.

As we can see from this figure, if the retained dimensionality of the decision-making space is too large, the algorithms tend to overfit and hence special consideration needs to be given to the choice of the numbers of projections used and the volume of training data.

7. Conclusion

In this work we presented a novel approach for equipping devices with limited computational capabilities with capabilities to quickly learn on-the-job and continuously improve over time in presence of spurious as well as a rather overwhelming number of errors. The approach is based on stochastic separation theorems [8,11,13,48,12,47] and the concentration of measure phenomena.

Our results demonstrate that the new capability can be delivered to the edge and deployed in a fully automated way, whereby a more sophisticated AI system monitors performance of a less powerful counterpart. The approach, for the first time uses 1NN integration rule for error correction, as opposed to mere disjunctions. The 1NN rule is justified by the dichotomy of high-dimensional datasets captured in Theorem 3 and Corollary 1. In addition, our results shed light on why few- and one-shot learning works. This new understanding can be used in the emerging frameworks for machine learning testing [55].

Experimentally, we investigated the sensitivity of the algorithm to change of its meta–parameters like the number of clusters and projections used. The results directly respond to the fundamental challenge of removing AI errors in industrial applications at minimal computational costs, and some elements of the theory underpin two US patents [40,39].

Theoretical results are illustrated with two industrial applications: performance monitoring of the CNC milling processes and edge-based object detection. An application field of the approach could be the class of randomized computational architectures such as stochastic configuration networks [52], and in particular the employment of the measure concentration effects for estimating their approximation convergence rates. The other relevant direction is to determine, for a given AI system, which signals in the system are most appropriate for the application of our algorithms. Recent work [2] showed that dimension of data representation in deep learning models may vary drastically depending on the layer where this representation is accessed. Therefore finding the best place to take information from for the purposes of few-shot AI correction is important. Present work shows that intrinsic dimension of data representation could be one of key factors in answering this question. Another interesting remaining question is how sensitive the approach is to label noise or Bayes errors. Answering these and other related questions in depth will be the subject of our future work.

CRediT authorship contribution statement

Ivan Y. Tyukin: Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review & editing. **Alexander N. Gorban:** Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review & editing. **Alistair A. McEwan:** Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review & editing. **Sepehr Meshkinfamfard:** Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review & editing. **Lixin Tang:** Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

A.N, I.T, and S.M were supported by Innovate UK Knowledge Transfer Partnership grant KTP010522, I.T. was supported by the UKRI Turing AI Acceleration Fellowship (EP/V025295/1), and L.T. was supported by a 111 Project (B16009). The work of I. T. at St Petersburg State Electrotechnical University was supported by the grant of the Russian Science Foundation (Project No. 19-19-00566, design of the algorithms and experiments), the work of A.N. and I.T. at Lobachevsky University was supported by the grant of the Ministry of Science and Higher Education of Russian Federation (Project No. 14.Y26.31.0022, theoretical analysis).

References

[1] A. Albergante, J. Bac, A. Zinovyev, Estimating the effective dimension of large biological datasets using fisher separability analysis, in: Proc. of the International Joint Conference on Neural Networks (IJCNN), 2019.
[2] A. Ansuini, A. Laio, J. Macke, and D. Zoccolan. Intrinsic dimension of data representations in deep neural networks. In Advances in Neural Information Processing Systems, pages 6111–6122, 2019..
[3] C. Bowman and P. Grindrod. Trust, limitation, conflation and hype. https://www.researchgate.net/publication/334425107_Trust_Limitation_Conflation_and_Hype, 2019..
[4] F. Censi, Calcagnini, E. Mattei, and A. Giuliani. System biology approach: Gene network analysis for muscular dystrophy. Methods in molecular biology (Clifton, N.J.), 1687: 75–89, 2018. ISSN 1064–3745. DOI: 10.1007/978-1-4939-7374-3_6..
[5] O. Chapelle, Training a support vector machine in the primal, Neural Comput. 19 (5) (2007) 1155–1178.
[6] T. Chen, I. Goodfellow, J. Shlens, Net2net, Accelerating learning via knowledge transfer, in: International Conference on Learning Representations (ICLR) 2016, 2015.
[7] G. Elekes, A geometric inequality and the complexity of computing volume, Discrete Comput. Geometry 1 (4) (1986) 289–292.
[8] A.N. Gorban, I.Y. Tyukin, Stochastic separation theorems, Neural Networks 94 (2017) 255–259, https://doi.org/10.1016/j.neunet.2017.07.014.
[9] A.N. Gorban, I.Y. Tyukin, Blessing of dimensionality: mathematical foundations of the statistical physics of data, Phil. Trans. R. Soc. A 376 (2018) 20170237, https://doi.org/10.1098/rsta.2017.0237.
[10] A.N. Gorban, I.Y. Tyukin, D.V. Prokhorov, K.I. Sofeikov, Approximation with random bases: Pro et contra, Inf. Sci. 364–365 (2016) 129–145, https://doi.org/10.1016/j.ins.2015.09.021.
[11] A.N. Gorban, I.Y. Tyukin, I. Romanenko, The blessing of dimensionality: Separation theorems in the thermodynamic limit, IFAC-PapersOnLine 49 (24) (2016) 64–69, https://doi.org/10.1016/j.ifacol.2016.10.755.
[12] A.N. Gorban, A. Golubkov, B. Grechuk, E.M. Mirkes, I.Y. Tyukin, Correction of AI systems by linear discriminants: Probabilistic foundations, Inf. Sci. 466 (2018) 303–322, https://doi.org/10.1016/j.ins.2018.07.040.
[13] A.N. Gorban, R. Burton, I. Romanenko, I.Y. Tyukin, One-trial correction of legacy AI systems and stochastic separation theorems, Inf. Sci. 484 (2019) 237–254, https://doi.org/10.1016/j.ins.2019.02.001.
[14] A.N. Gorban, B. Grechuk, I.Y. Tyukin. Augmented artificial intelligence. arXiv preprint arXiv:1802.02172, 2018..
[15] A.N. Gorban, V.A. Makarov, I.Y. Tyukin, The unreasonable effectiveness of small neural ensembles in high-dimensional brain, Phys. Life Rev. (2018), https://doi.org/10.1016/j.plrev.2018.09.005.
[16] A.N. Gorban, V.A. Makarov, I.Y. Tyukin, High-dimensional brain in a high-dimensional world: Blessing of dimensionality, Entropy 22 (1) (2020) 82, https://doi.org/10.3390/e22010082.
[17] B. Grechuk, A.N. Gorban, General stochastic separation theorems with optimal bounds, Neural Networks 138 (2021) 33–56, https://doi.org/10.1016/j.neunet.2021.01.034.
[18] M. Gromov, Isoperimetry of waists and concentration of maps, GAFA, Geomteric Funct. Anal. 13 (2003) 178–215.
[19] G. Hains, A. Jakobsson, Y. Khmelevsky, Towards formal methods and software engineering for deep learning: Security, safety and productivity for DL systems development, in: 2018 Annual IEEE International Systems Conference (SysCon), IEEE, 2018, pp. 1–5, https://doi.org/10.1109/SYSCON.2018.8369576.
[20] L.K. Hansen, P. Salamon, Neural network ensembles, IEEE Trans. Pattern Anal. Mach. Intell. 12 (10) (1990) 993–1001.
[21] P. Hart, The condensed nearest neighbor rule (corresp.), IEEE Trans. Inform. Theory 14 (3) (1968) 515–516.
[22] S.C. Hicks, F.W. Townes, M. Teng, R.A. Irizarry. Missing data and technical variability in single-cell RNA-sequencing experiments. Biostatistics, 19 (4): 562–578, 11 2017. ISSN 1465–4644. DOI: 10.1093/biostatistics/kxx053..
[23] T.K. Ho, Random decision forests, in: Proc. of the 3rd International Conference on Document Analysis and Recognition, 1995, pp. 993–1001.
[24] T.K. Ho, The random subspace method for constructing decision forests, IEEE Trans. Pattern Anal. Mach. Intell. 20 (8) (1998) 832–844.
[25] P. Kainen, V. Kůrková, Quasiorthogonal dimension of Euclidean spaces, Appl. Math. Lett. 6 (3) (1993) 7–10.
[26] P.C. Kainen, Utilizing geometric anomalies of high dimension: When complexity makes computation easier, in: Computer Intensive Methods in Control and Signal Processing, Springer, 1997, pp. 283–294.
[27] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A.A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, R. Hadsell, Overcoming catastrophic forgetting in neural networks, Proc. Nat. Acad. Sci. 114 (13) (2017) 3521–3526.
[28] N. Krumm, P.H. Sudmant, A. Ko, B.J. O'Roak, M. Malig, B.P. Coe, A.R. Quinlan, D.A. Nickerson, E.E. Eichler, Copy number variation detection and genotyping from exome sequence data, Genome Research, 10889051 228 (Aug 2012) 1525–1532, https://doi.org/10.1101/gr.138115.112.
[29] A. Kuznetsova, S. Hwang, B. Rosenhahn, L. Sigal, Expanding object detector's horizon: Incremental learning framework for object detection in videos, in: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 28–36.
[30] A. Lasemi, D. Xue, P. Gu, Recent development in CNC machining of freeform surfaces: A state-of-the-art review, Comput. Aided Des. 42 (7) (2010) 641–654.
[31] N. Li, D.W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, A.R. Girard, Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems, IEEE Trans. Control Syst. Technol. 26 (5) (Sep. 2018) 1782–1797, https://doi.org/10.1109/TCST.2017.2723574.
[32] H. Liang, B. Tsui, H. Ni, C. Valentim, S. Baxter, G. Liu, W. Cai, D. Kermany, K. Sun, J. Chen, L. He, J. Zhu, P. Tian, H. Shao, L. Zheng, R. Hou, S. Hewett, G. Li, P. Liang, X. Zang, Z. Zhang, L. Pan, H. Cai, R. Ling, S. Li, Y. Cui, S. Tang, H. Ye, X. Huang, W. He, W. Liang, Q. Zhang, J. Jiang, W. Yu, J. Gao, W. Ou, Y. Deng, Q. Hou, B. Wang, C. Yao, Y. Liang, S. Zhang, Y. Duan, R. Zhang, S. Gibson, C. Zhang, O. Li, E. Zhang, G. Karin, N. Nguyen, X. Wu, C. Wen, J. Xu, W. Xu, B. Wang, W. Wang, J. Li, B. Pizzato, C. Bao, D. Xiang, W. He, S. He, Y. Zhou, W. Haw, M. Goldbaum, A. Tremoulet, C.-N. Hsu, H. Carter, L. Zhu, K. Zhang, H. Xia, Evaluation and accurate diagnoses of pediatric diseases using artificial intelligence, Nat. Med. 25 (2019) 433–438, https://doi.org/10.1038/s41591-018-0335-9.
[33] H. Lu, K.N. Plataniotis, A.N. Venetsanopoulos, A survey of multilinear subspace learning for tensor data, Pattern Recogn. 44 (7) (2011) 1540–1551.
[34] D. Meltz, H. Guterman, Functional safety verification for autonomous ugvs–methodology presentation and implementation on a full-scale system, IEEE Trans. Intelligent Veh. 4 (3) (Sep. 2019) 472–485, https://doi.org/10.1109/TIV.2019.2919460.
[35] S. Mashkinfamfard. Streets of Montreal dataset, 2020. https://github.com/Sep-AI/HaarCascade_Vs_MTCNN..
[36] I. Misra, A. Shrivastava, M. Hebert, Semi-supervised learning for object detectors from video, in: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3594–3602.
[37] L. Pratt, Discriminability-based transfer between neural networks, Advances in Neural Information Processing Systems 5 (1992) 204–211.
[38] A. Prest, C. Leistner, J. Civera, C. Schmid, V. Ferrari, Learning object class detectors from weakly annotated video, in: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 3282–3289.
[39] I. Romanenko, I. Tyukin, A. Gorban, K. Sofeikov. Method of image processing. US patent US10062013B2, August, 28 2018. https://patents.google.com/patent/US10062013B2/en..
[40] I. Romanenko, A. Gorban, I. Tyukin. Image processing. US patent US10489634B2, November, 26 2019. https://patents.google.com/patent/US20180089497A1/en..
[41] K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. In International Conference on Learning Representations, 2015. arXiv:1409.1556..
[42] J. Snell, K. Swersky, R. Zemel. Prototypical networks for few-shot learning. In Advances in neural information processing systems, pages 4077–4087, 2017..

[43] N. Sompairac, P.V. Nazarov, U. Czerwinska, L. Cantini, A. Biton, A. Molkenov, Z. Zhumadilov, E. Barillot, F. Radvanyi, A.N. Gorban, U. Kairov, A. Zinovyev. Independent component analysis for unraveling the complexity of cancer omics datasets. Int. J. Mol. Sci., 20 (18): 4414, Sep 2019. ISSN 1422–0067. DOI: 10.3390/ijms20184414..
[44] E. Strickland, IBM Watson, heal thyself: How IBM overpromised and underdelivered on AI health care, IEEE Spectr. 56 (4) (2019) 24–31, https://doi.org/10.1109/MSPEC.2019.8678513.
[45] S. Sun. CNC mill tool wear dataset, 2018. https://www.kaggle.com/shasun/tool-wear-detection-in-cnc-mill..
[46] A. Takács, I. Rudas, D. Bösl, T. Haidegger, Highly automated vehicles and self-driving cars [industry tutorial], IEEE Robotics Autom. Magazine 25 (4) (Dec 2018) 106–112, https://doi.org/10.1109/MRA.2018.2874301.
[47] I.Y. Tyukin, A.N. Gorban, S. Green, D. Prokhorov, Fast construction of correcting ensembles for legacy artificial intelligence systems: Algorithms and a case study, Inf. Sci. 485 (2019) 230–247, https://doi.org/10.1016/j.ins.2018.11.057.
[48] Tyukin I.Y., Gorban A.N., Sofeikov K., Romanenko I, Knowledge transfer between artificial intelligence systems. Frontiers of Neurorobotics, 12, Article 49, 2018. https://doi.org/10.3389/fnbot.2018.00049..
[49] V. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, 2000.
[50] R. Izmailov, V. Vapnik, Knowledge transfer in SVM and neural networks, Annals of Mathematics and Artificial Intelligence (2017) 1–17.
[51] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, D. Wierstra. Matching networks for one shot learning. In Advances in neural information processing systems, pages 3630–3638, 2016..
[52] D. Wang, M. Li, Stochastic configuration networks: Fundamentals and algorithms, IEEE Trans. Cybern. 47 (10) (2017) 3466–3479.
[53] Y. Wang, Q. Yao, J. Kwok, L.M. Ni, Generalizing from a few examples: A survey on few-shot learning, ACM Computing Surveys (CSUR) 53 (3) (2020) 1–34.
[54] C. Zhang, S. Bengio, M. Hardt, B. Recht, O. Vinyals. Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530, 2016..
[55] J. Zhang, M. Harman, L. Ma, Y. Liu, Machine learning testing: Survey, landscapes and horizons, IEEE Trans. Software Eng. (2020).
[56] S. Zheng, Y. Song, T. Leung, I. Goodfellow, Improving the robustness of deep neural networks via stability training, in: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, https://arxiv.org/abs/1604.04326.