Incremental Object Recognition in Robotics with Extension to New Classes in Constant Time

Raffaello Camoriano∗†‡, Giulia Pasquale∗†, Carlo Ciliberto∗, Lorenzo Natale†, Lorenzo Rosasco∗‡ and Giorgio Metta†

Abstract—We consider object recognition in the context of lifelong learning, where a robotic agent learns to discriminate between a growing number of object classes as it accumulates experience about the environment. We propose an incremental variant of the Regularized Least Squares for Classification (RLSC) algorithm, and exploit its structure to seamlessly add new classes to the learned model. The presented algorithm addresses the problem of having unbalanced proportion of training examples per class, which occurs when new objects are presented to the system for the first time.

We evaluate our algorithm on both a machine learning benchmark dataset and a challenging object recognition task in a robotic setting. Empirical evidence on both problems shows that our approach is significantly faster than its batch counterparts while achieving comparable or better classification performance when classes are unbalanced.

I. INTRODUCTION

In order for robotic agents to operate in unstructured environments, several advanced perceptual capabilities are required. In this work, we focus on visual object recognition. In particular, we consider the learning algorithms involved in the object recognition process, framed as a supervised classification problem with multiple classes. When performing object recognition, the robot typically acquires labeled pictures of new objects presented to it and learns a predictive model based on them. It then employs the learned model for predicting the labels of new, previously unseen objects. This strategy is usually implemented with batch learning algorithms, given a fixed number of examples and classes. Once a batch model has been trained, it can only be deployed for prediction, without taking advantage of new supervised data points which might become available, it cannot adapt to changes in the environmental properties or, most importantly, include novel object classes. This drawback represents an obstacle for robots to continuously expand their predictive capabilities in a lifelong learning setting [1]. In the ideal reference scenario, the classifier can be updated with new points, allowing for improved accuracy and adaptation to concept drift, and can also be extended with new classes. Yet, re-training a batch model as soon as a new example becomes available is out of the question, due to the excessive computational and memory requirements needed for training set storage and learning algorithm execution. The proposed method exploits the structural properties of Regularized Least Squares for Classification (RLSC) (see [2], [3]) to allow for incremental model updates with class extension. By doing so, fixed update complexity is maintained, which is the main requirement for real-time computations. Indeed, this is essential for a robot which is expected to promptly react to external stimuli. Furthermore, we provide a fully incremental recoding strategy for significantly improving accuracy when class imbalance is significant, as is often the case for robotic object recognition acquisition procedures.

The paper is organized as follows: First, in Section II, we introduce the unbalanced classification setting and present several possible learning approaches. Then, in Section III the RLS algorithm is presented, together with its recursive version. In Section IV we introduce our method, focusing on incremental class extension, model selection and recoding. Finally, Section V illustrates the results of accuracy and computational time benchmarking experiments, including a real-world robotic scenario implemented on the iCub humanoid robot [4].

II. THE CLASSIFICATION PROBLEM WITH UNBALANCED DATA

In this section we introduce the learning framework for classification, describe the disrupting effects of imbalance of class labels and discuss potential solutions to mitigate this issue in practice. We refer the reader to [5] for more details about the Statistical Learning Theory for classification problems.

For the sake of simplicity, in the following we discuss the classification problem in the binary setting. At the end of this Section we generalize our analysis to multiclass.

A. The Optimal Bayes Classifier and the Least Squares Surrogate

Let us consider a binary classification problem where input-output examples are sampled randomly according to a distribution \( p \sim \mathcal{X} \times \{-1, 1\} \). Our goal is to learn the best function \( b_\ast : \mathcal{X} \to \{-1, 1\} \), minimizing the overall expected classification error

\[
 b_\ast = \arg\min_{b: \mathcal{X} \to \{-1, 1\}} \int_{\mathcal{X} \times \{-1, 1\}} I_{b(x) \neq y} dp(x, y) 
\]

from a finite set of observations \( \{x_i, y_i\}_{i=1}^n \), \( x_i \in \mathcal{X}, y_i \in \{-1, 1\} \). Here \( I_a \) denotes the binary function taking value
0 if $a = 0$ and 1 otherwise. The solution to (1) is called the \textit{optimal Bayes classifier} and it can be easily shown that it is of the form

$$b_*(x) = \begin{cases} 1 & \text{if } \rho(y = 1|x) > \rho(y = -1|x) \\ -1 & \text{otherwise} \end{cases}.$$ \hspace{1cm} (2)

When only a finite and small number of training examples is provided, estimates of the distribution $\rho(x, y)$ or $\rho(y|x)$ cannot be computed accurately, and minimizing Eq. (1) poses a challenging problem. To this end, in practice, a so-called \textit{surrogate} problem (see [5], [6]) is often adopted to simplify the optimization problem and asymptotically recover the optimal Bayes classifier. In this sense, one well-known surrogate approach is to consider the least-squares expected risk minimization

$$f_* = \arg\min_{f: \mathcal{X} \to \mathbb{R}} \int_{\mathcal{X} \times \{-1, 1\}} (y - f(x))^2 d\rho(x, y).$$ \hspace{1cm} (3)

Indeed, it is easy to see that the solution to Eq. (3) recovers the optimal Bayes classifier. To see this, let us assume that for $\rho$ the Bayes’ rule holds, namely that $\rho(x, y) = \rho(y|x)\rho(x)$. Then we have

$$\int_{\mathcal{X} \times \{-1, 1\}} (y - f(x))^2 d\rho(x, y) =$$

$$\int_{\mathcal{X}} (y - f(x))^2 d\rho(y|x) d\rho(x) =$$

$$\int_{\mathcal{X}} [(1 - f(x))^2 \rho(y = 1|x) +$$

$$(f(x) + 1)^2 \rho(y = -1|x)] d\rho(x),$$ \hspace{1cm} (4)

from which follows that, for every $x \in \mathcal{X}$, the minimizer of Eq. (4) is $f_*(x) = 2\rho(y = 1|x) - 1 = \rho(y = 1|x) - \rho(y = -1|x)$. As a consequence, the optimal Bayes classifier can be written as $b_*(x) = \text{sign}(f_*(x))$, since $f_*(x) > 0$ if and only if $\rho(y = 1|x) > \rho(y = -1|x)$.

\textbf{Empirical Setting.} When solving the problem in practice we are provided with a finite set $\{x_i, y_i\}_{i=1}^n$ of training examples. In these settings the typical approach is to solve the empirical risk minimization

$$\hat{f} = \arg\min_{f: \mathcal{X} \to \mathbb{R}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2 + R(f),$$ \hspace{1cm} (5)

where $R$ is a so-called \textit{regularizer} preventing the solution $\hat{f}$ to overfit. Indeed, it can be shown [5], [7] that under mild assumptions on the distribution $\rho$, it is possible for $\hat{f}$ to converge in probability to the ideal $f_*$ as the number of training points grows indefinitely. A variety of methods exists to deal with Eq. (5) and indeed Sec. III provides some background in this sense.

\textbf{B. The Effect of Unbalanced Data}

The classification rule in Eq. (2) associates every new example $x \in \mathcal{X}$ to the class $y$ whose likelihood $\rho(y|x)$ is the highest. However, in settings where the two classes are not balanced (e.g. $\rho(y = 1) >> \rho(y = -1)$) this could lead to unexpected and undesired behaviors. To see this, let us denote $\gamma = \rho(y = 1)$ and notice that, combining Eq. (2) with Bayes’ rule, we have that class 1 is associated to an example $x$ whenever

$$\rho(x|y = 1) > \rho(x|y = -1)\frac{(1-\gamma)}{\gamma}.$$ \hspace{1cm} (6)

When $\gamma$ is close to one of its extremal values 0 or 1 (hence $\rho(y = 1) >> \rho(y = -1)$ or vice-versa), one class becomes clearly preferred with respect to the other and is almost always selected.

In Fig. 1 we report an example of the effect of unbalanced data by showing how the decision boundary (white dashed curve) of the optimal Bayes classifier from Eq. (2) varies as $\gamma$ takes values from 0.5 (balanced case) to 0.9 (very unbalanced case). As it can be noticed, while the classes maintain the same form, the decision boundary is remarkably affected by the value of $\gamma$.

Clearly, this effect could be desirable in some settings (i.e. when the overall error rate is indeed a concern). However, in robotics setting this is critically suboptimal for two reasons: 1) We would like the robot to recognize with high accuracy even objects that are less common to be seen. 2) In incremental settings, whenever a novel object is observed for the first time, only few training examples are available (in the extreme case even just one!) and we need a loss weighting fairly also underrepresented classes.

![Fig. 1: Bayes decision boundaries for standard (dashed white line) and weighted (dashed black line) binary classification loss and multiple values of $\gamma = \rho(y = 1)$ from 0.5 to 0.9. The data points are sampled according to Gaussian distributions $\rho(x|y = 1) \sim N(\mu_1, \sigma_1)$ and $\rho(x|y = -1) \sim N(\mu_2, \sigma_2)$ with $\mu_1 = (-1, 0)^\top$, $\mu_2 = (1, 0)^\top$, $\sigma_1 = 1$ and $\sigma_2 = 0.3$. As it can be noticed, the two boundaries coincide when $\gamma = 0.5$ (balanced data) while they lead to different regions when the data distribution becomes more unbalanced.](image-url)
C. Rebalancing the Loss

In this paper, we consider a general approach to “rebalancing” the classification loss of the standard learning problem of Eq. (1). We begin by noticing that in the balanced setting, namely for \( \gamma = \frac{1}{2} \), the classification rule at Eq. (6) is equivalent to \( \rho(x|y = 1) > \rho(x|y = -1) \). We want to slightly modify the classification loss in order to recover this same rule also in unbalanced settings. To do so, we propose to apply a weight \( w(y) \in \mathbb{R} \) to the loss \( \mathbb{I}_{b(x) \neq y} \) in Eq. (1), obtaining the weighted classification problem

\[
\hat{b}_w^* = \arg\min_{b:X \to \{-1,1\}} \int_{X \times \{-1,1\}} w(y)\mathbb{I}_{b(x) \neq y}d\rho(x, y) \tag{7}
\]

Analogously to the non-weighted case, the solution to this problem is

\[
b_w^*(x) = \begin{cases} 
1 & \text{if } \rho(y = 1|x)w(1) > \rho(y = -1|x)w(-1) \\
-1 & \text{otherwise}
\end{cases} \tag{8}
\]

In this work we take the weights \( w \) to be positive \( w(1) = \frac{1}{\rho(y = 1)} \) and \( w(-1) = \frac{1}{\rho(y = -1)} \). Indeed, from the fact that \( \rho(y|x) = \rho(x|y)/\rho(x) \) we have that the rule at (9) is equivalent to

\[
b_w^*(x) = \begin{cases} 
1 & \text{if } \rho(y|x) > \rho(x|y) \text{ and } \rho(x|y) > 1 \\
-1 & \text{otherwise}
\end{cases} \tag{9}
\]

that corresponds to the (unbalanced) optimal Bayes classifier in the case \( \gamma = 0.5 \), as desired.

Fig. [1] compares the unbalanced and rebalanced optimal Bayes classifiers for different values of \( \gamma \). Notice that rebalancing leads to solutions that are invariant to the value of \( \gamma \) (compare the black decision boundary with the white one).

D. Rebalancing and Recoding the Least-Squares Loss

Interestingly, the strategy of changing the weight of the classification error loss can be naturally extended to the least-squares surrogate. If we consider the weighted least-squares problem,

\[
f_w^* = \arg\min_{f:X \to \mathbb{R}} \int_{X \times \{-1,1\}} w(y)(y - f(x))^2d\rho(x, y), \tag{10}
\]

we can again recover the surrogate rule \( b_w^* = \text{sign} \circ f_w^* \) observed in the non-weighted setting. Indeed, by direct calculation it can be shown that Eq. (10) has solution

\[
f_w^*(x) = \frac{\rho(y = 1|x)w(1) - \rho(y = -1|x)w(-1)}{\rho(y = 1|x)w(1) + \rho(y = -1|x)w(-1)} \tag{11}
\]

and if we take the weights to be positive \( w(y) > 0 \) we have that the denominator of Eq. (11) is always positive and therefore \( \text{sign}(f_w^*(x)) = 1 \) if and only if \( \rho(y = 1|x)w(1) > \rho(y = -1|x)w(-1) \).

Coding. Another approach to recover the rebalanced optimal Bayes classifier via the least-squares loss is to apply a suitable coding function to the class labels \( y = \{-1,1\} \), namely

\[
f_w^* = \arg\min_{f:X \to \mathbb{R}} \int_{X \times \{-1,1\}} (c(y) - f(x))^2d\rho(x, y), \tag{12}
\]

where \( c : \{-1,1\} \to \mathbb{R} \) maps the labels \( y \) into suitable codes \( c(y) \in \mathbb{R} \). Analogously to the unbalanced (and uncoded) case, the solution to Eq. (12) can be shown to be \( f_w^*(x) = c(1)(\rho(y = 1|x) + c(-1)(\rho(y = -1|x) \) and therefore we can just take \( c(y) = w(y) \) to recover the rebalanced optimal Bayes classifier.

Note. We care to point out that, while the coding strategy is asymptotically equivalent to the reweighted least-squares, in empirical settings they are substantially different. In particular, in the incremental context described in Sec. III we show that it is possible to derive a recoded recursive least-squares algorithm, while the same is not possible for the reweighted case. This technical difference is also reflected empirically in the analysis reported in Sec. V.

E. Multiclass Rebalancing and Recoding

Here we extend the weighting and coding strategies proposed for the binary setting to multiclass classification. In the multiclass setting, the optimal Bayes decision rule corresponds to the function \( b_{\rho} : X \to \{1, \ldots, T\} \), assigning a label \( i \in \{1, \ldots, T\} \) to the sample \( x \in X \) such that \( \rho(y = i|x) > \rho(y = j|x) \) \( \forall j \neq i \). Consequently, the rebalanced decision rule would become \( \rho(y = i|x)w(i) > \rho(y = j|x)w(j) \) \( \forall j \neq i \), with \( w : \{1, \ldots, T\} \to \mathbb{R} \) determining a weight for each class. Generalizing the binary case, in this work we consider \( w(i) = \frac{1}{\rho(y=i)} \) for each \( i \in \{1, \ldots, T\} \).

The surrogate least-squares classification setting is recovered by adopting a 1-vs-all approach, namely solving the vector-valued problem

\[
f_* = \arg\min_{f:X \to \mathbb{R}^T} \int_{X \times \{-1,1\}} \|y - f(x)\|^2d\rho(x, y), \tag{13}
\]

where \( y \in \mathbb{R}^T \) is a vector of the canonical basis \( \{e_1, \ldots, e_T\} \) of \( \mathbb{R}^T \) whose entries are all 0 except for the one corresponding to the correct class which is equal to 1. Analogously to the derivation at Eq. (II-A), the solution to this problem can be easily shown to correspond to the is the vector-valued predictor \( f_*(x) = (\rho(y = 1|x), \ldots, \rho(y = T|x))^T \). Consequently we recover the optimal Bayes classifier according to the operation

\[
b_*(x) = \arg\max_{i=1,\ldots,T} f_*(x)_i. \tag{14}
\]

where \( f_*(x)_i \) denote the \( i \)-th entry of the vector \( f_*(x) \in \mathbb{R}^T \).

The extension of coding and weighting approaches to this setting follow analogously to the binary setting discussed in Sec. II-D. In particular, the coding function \( c : \{e_1, \ldots, e_T\} \to \mathbb{R} \) consists in mapping a vector of the basis \( e_i \) to \( c(e_i) = \frac{1}{\rho(y=i)}e_i \).

Note. In the previous sections we presented the analysis on the binary case by considering a \( \{-1,1\} \) coding for the class labels. Indeed this allows for a more clear introduction of the problem since we need to solve a single least squares problem to recover the optimal Bayes classifier. Alternatively, we could follow the approach introduced in this section and, adopt the labels \( c_1 = (1,0) \) and \( c_2 = (0,1) \) for the
two classes and training two distinct classifiers, one per class, taking the argmax of their scores to be the predicted class. The two approaches are clearly equivalent since in both cases the Bayes classifier is based on the inequality $\rho(y = 1|x) > \rho(y = -1|x)$ or $\rho(y = 1|x) > \rho(y = 2|x)$.

III. INCREMENTAL REGULARIZED LEAST SQUARES

In this section we introduce the incremental Regularized Least Squares algorithm used for classification. We first review the standard Regularized Least Squares setting and then proceed by describing the recursive algorithm used for incremental updates of the least-squares solution.

A. Regularized Least Squares for Classification

In Sec. II we have mentioned that in empirical settings the empirical least-squares problem at (5) can be addressed to learn a good estimate of the actual solution. To this end, let us consider a finite training set $\{x_i, y_i\}_{i=1}^n$ such that the inputs $x_i \in \mathcal{X} = \mathbb{R}^d$ and the labels $y_i \in \{1, \ldots, c\}$ are represented as vectors as described in Sec. II-E. Then, if we assume a linear model for the classifier $f$, we can rewrite Eq. (5) in compact matrix notation as

$$W_\star = \arg\min_{W \in \mathbb{R}^{d \times d}} \|Y - XW\|^2 + \lambda\|W\|^2$$

(15)

where $\lambda > 0$ is a so-called regularization parameter $X \in \mathbb{R}^{n \times d}$ and $Y \in \mathbb{R}^{n \times \mathcal{Y}}$ are the matrices whose rows correspond respectively to the training examples, and $\|\cdot\|^2$ denotes the squared Frobenius norm (i.e. the sum of the squared entries of a matrix).

It is well known that the solution to Eq. (15) is of the form

$$W_\star = (X^T X + \lambda I_d)^{-1} X^T Y$$

(16)

where $I_d$ is the $d \times d$ identity matrix.

Prediction. According to the rule introduced in Sec. II-E whenever a new example $x \in \mathbb{R}^d$ is provided, the classification prediction of the regularized least squares classifier is then obtained by taking the

$$\arg\max_{i=1, \ldots, \mathcal{Y}} f(x)_i = x^T W_\star^{(i)}$$

(17)

with $W_\star^{(i)} \in \mathbb{R}^d$ denoting the $i$-th column of $W_\star$.

B. Recursive Regularized Least Squares

We now review the Recursive RLS (RRLS) algorithm [8], consisting in an efficient update rule for the solution of Eq. (16), which allows to compute it incrementally whenever new examples $(x_i, y_i)$ are provided to the system.

We consider a setting where data points arrive to the learning system one at the time and at each time instant $k$ we want to compute the updated solution to Eq. (16)

$$W_k = (X_k^T X_k + \lambda I_d)^{-1} X_k^T Y_k.$$  

Here $X_k \in \mathbb{R}^{k \times d}$ and $Y_k \in \mathbb{R}^{k \times \mathcal{Y}}$ are the matrices containing the training dataset observed up to iteration $k$. In particular we have that at each step $k$

$$X_k = [X_{k-1}^T, x_k]^T, \quad Y_k = [Y_{k-1}^T, y_k]^T.$$

Therefore, if we denote denote $A_k = X_k^T X_k + \lambda I_d$ and $b = X_k^T Y_k$, we have the recursive formulas

$$A_k = X_{k-1}^T X_{k-1} + x_k^T x_k + \lambda I_d$$

$$A_{k-1} + x_k^T x_k + \lambda I_d$$

(18)

and

$$b_k = X_k^T Y_k = X_{k-1}^T Y_{k-1} + x_k y_k^T.$$

(19)

These recursive formulas for $A_k$ and $b_k$ allow for the model of $W_k$ to be updated from the previous state, however an inversion of $A_k$ is still required at each step since $W_k = A_k^{-1} b_k$. Matrix inversion has a time complexity of $O(d^3)$, which is often not feasible for real-time applications.

A solution to this problem is to consider the Cholesky decomposition of $A_k = R_k^T R_k$ (see [9]) and perform a rank one update of $R_k$. Indeed, the inversion of a matrix for which the Cholesky decomposition is known can be performed in $O(d^2)$. The rank-one update of a Cholesky decomposition can be performed in $O(d^2)$ as well so the total cost of each iteration would amount to $O(d^2)$ rather than $O(d^3)$ for a remarkable computational speed-up.

Now, the problem of updating the Cholesky decomposition of $A_k$ is

$$A_k = R_k^T R_k = A_{k-1} + x_k^T x_k$$

$$R_{k-1} + x_k^T x_k, \quad (20)$$

where $R$ is full rank, and $R_0 = \sqrt{\lambda I_d}$. If we define

$$\tilde{R}_{k-1} = \left[ \begin{array}{c} \tilde{R}_{k-1}^{-1} \\ x_k \end{array} \right] \in \mathbb{R}^{(d+1) \times d}, \quad (21)$$

we can write $A_k = \tilde{R}_k^T \tilde{R}_k^{-1}$. The Cholesky update procedure, based on Givens rotations [10], can be used to compute $R_k$ from $\tilde{R}_{k-1}$ with $O(d^2)$ time complexity. Several implementations of this procedure are available; for our experiments we used the CHOLUPDATE routine in MATLAB.

IV. INCREMENTAL MULTICLASS CLASSIFICATION WITH CLASS EXTENSION AND RECODING

In this Section, we present a simple approach to RRLS incremental multiclass classification where we account for the possibility to extend the number of classes incrementally and apply the recoding approach introduced in Sec. II.

A. Class Extension

We introduce a modification of RRLS for classification, allowing for the extension of the number of classes in the model. Given the methods described in Sec. III we shall now consider the setting in which the number of classes $T_k$ at step $k$ may change. At each step $k$, two cases may arise:
1) The new sample \((x_k, y_k)\) belongs to a known class, this meaning that \(y_k \in \mathbb{R}^{T_k-1}\) and \(T_k = T_{k-1}\).

2) \((x_k, y_k)\) belongs to a previously unknown class, implying that \(y_k \in \mathbb{R}^{T_k}\) and \(T_k = T_{k-1} + 1\).

In the first case, the update rules for \(A_k, b_k\) and \(W_k\) explained in Section III-B can be directly applied. In the second case, the update of \(b_k\) changes, since its size depends on the current number of classes \(T_k\). Nevertheless, the computation of \(W_k\) stays the same, as it implies full forward-backward substitution at each step, given the updated \(R_k\) and \(b_k\). Thus, in case 2, the update rule for \(b_k \in \mathbb{R}^{d \times T_k}\) becomes

\[
b_k = \begin{bmatrix} 0 \\ b_{k-1} \\ \vdots \\ 0 \end{bmatrix} + x_k^T y_k. \tag{22}
\]

Note that since we have restricted the classification problem to 1-vs-all, the term \(x_k^T y_k\) will only modify the newly added column of zeros in \(b_k\).

With this strategy, it is possible to extend the classification capabilities of the incremental learner during online operation, without re-training it from scratch. The dataset imbalance problem and the choice of the regularization parameter \(\lambda\) are dealt with in the following subsections.

### B. Incremental Recoding

We want to adapt the recoding strategy introduced in Sec. [III] to the incremental Regularized Least Squares setting. The only difference of this approach with respect to the standard incremental RLS is that at each iteration \(b_k\) is now obtained as

\[
b_k = X_k^T \Gamma_k Y_k, \tag{23}
\]

with \(\Gamma_k \in \mathbb{R}^{k \times k}\) a diagonal matrix whose entries, \((\Gamma_k)_{i,j}\) correspond to the codes that we are imposing on the lass labels \(Y_k\). Following the method in Sec. [IV-E] we assign to each \(i\) the ratio \(k_i / k_c\), where \(k_c\) is the number of training examples observed of the class \(c_i\) to which the \(i\)-th example \(x_i\) belongs. Indeed \(k_i / k_c\) is a naive estimate of the quantity \(1/\rho(y = i)\) used in the ideal setting.

Eq. (23) seem to suggest that every time \(\Gamma_k\) changes, \(b_k\) should be computed from scratch. This would lead to \(O(dkT)\) computations at each iteration, which is undesirable due to the dependency on \(k\). However, by consider the special binary structure of \(Y_k\), it can be shown by linear algebra that actually for each \(k\), the final “recoded” classifier is of the form \(W_{\text{coded}} = \Gamma W_*\), where \(W_*\) is the solution of Eq. (16). This means that for every input \(x\), the corresponding class is predicted by computing the argmax

\[
\text{argmax}_{c=1,\ldots,T} \ f_{\text{coded}}(x) = x^T \ W_* = \frac{k}{k_c} x^T \ W_{\text{coded}}. \tag{24}
\]

Therefore to extend RRLS to the recoding setting it is sufficient to suitably weight the prediction scores of the vector-valued classifier obtained by standard RRLS before taking the argmax. This preprocessing does not impact in any way the computational performances.

### Coding as a Regularization parameter

We note that depending on the amount of training examples, the estimate \(k_i / k_c\) of invers distribution \(1/\rho(y = i)\) could be very inaccurate. Therefore, in order to mitigate the influence of this possible hindrance, here we propose to tune a regularization parameter \(\alpha \in [0,1]\) controlling the power of the coding, namely using \(b_k = X_k^T \Gamma_k^\alpha Y_k\). Such regularizer would be set to values closer and closer to 1 as we get more examples and a better estimate of \(\rho(y = i)\).

### Incremental Reweighting

We note here that it is not possible to design an incremental reweighting algorithm via the same strategy discussed in this work. Indeed, the solution of the reweighted RLS is of the form

\[
W_k = (X_k^T \Gamma_k X_k + \lambda I_d)^{-1} X_k^T \Gamma_k Y_k. \tag{25}
\]

However, the term \(\Gamma_k\), which changes at every step, does not allow to derive a rank-one update rule as it is the case for standard RRLS. Therefore it is not possible in general to derive a recursive methods in this setting.

### C. Incremental Model Selection

We now discuss the choice of the optimal regularization parameter \(\lambda^*\) in the incremental learning setting. Indeed, in general \(\lambda^*\) might change as \(n\) grows. Still, it is not possible for a model which is incremental in \(n\) to change \(\lambda\) at each step as in the batch case. A practical solution is to consider an array of independent models in parallel, each with a different \(\lambda\) chosen in a predefined range, and perform cross validation incrementally to choose the best model at each step. This incremental cross validation procedure can be performed efficiently, by keeping a separate balanced validation set on which to evaluate the accuracy of the models at every step and select the one yielding the highest accuracy.

### V. EXPERIMENTS

In this Section, we present two experimental evaluations of the proposed incremental approach, comparing it with other state of the art methods in terms of predictive and computational performance. We consider the datasets reported in Table [I].

#### A. Accuracy Benchmarking on MNIST

We first compare the incremental RLS multiclass classification algorithm, with and without recoding, and the batch version with exact loss rebalancing. The benchmark learning

### Table I: Specifications of the benchmark datasets used in the experiments.

| Dataset      | \(n\) | \(n_{\text{test}}\) | \(d\) | \(T\) |
|--------------|------|-------------------|------|------|
| MNIST        | 60000| 10000             | 784  | 10   |
| iCubWorld28  | 19669| 19330             | 60000| 1000 | 784  | 28   |
TABLE II: Test accuracy for the exactly rebalanced batch (Rebal.), recoding incremental (Recod.) and naïve incremental (Naïve) RLSC on the MNIST dataset for different unbalancing rates of the “9” digit.

| n | n_{bal} | n_{imb} | Rebal. | Acc. | Recod. | Acc. | Naïve | Acc. |
|---|---------|---------|--------|------|--------|------|-------|------|
| 1000 | 110 | 10 | 80.5 ± 2.8% | 79.6 ± 1.9% | 77.3 ± 1.1% | 77.0 ± 0.6% | 77.0 ± 0.6% | 77.0 ± 0.6% |
| 1000 | 109 | 20 | 82.3 ± 0.9% | 80.9 ± 1.2% | 76.6 ± 1.2% | 78.2 ± 0.7% | 78.2 ± 0.7% | 78.2 ± 0.7% |
| 1000 | 107 | 40 | 83.4 ± 0.6% | 82.9 ± 0.6% | 79.6 ± 0.9% | 80.2 ± 0.3% | 80.2 ± 0.3% | 80.2 ± 0.3% |
| 1000 | 100 | 100 | 83.9 ± 0.7% | 83.9 ± 0.7% | 83.9 ± 0.7% | 83.9 ± 0.7% | 83.9 ± 0.7% | 83.9 ± 0.7% |

TABLE III: Test accuracy for the exactly rebalanced batch (Rebal.), recoding incremental (Recod.) and naïve incremental (Naïve) RLSC on the iCubWorld28 dataset for different unbalancing rates of the 28th class.

| n | n_{bal} | n_{imb} | Rebal. | Acc. | Recod. | Acc. | Naïve | Acc. |
|---|---------|---------|--------|------|--------|------|-------|------|
| 19669 | >767 | 11 | 77.9 ± 0.3% | 79.1 ± 0.3% | 77.8 ± 0.0% | 78.0 ± 0.2% | 78.0 ± 0.2% | 78.0 ± 0.2% |
| 19669 | >767 | 19 | 79.0 ± 0.5% | 79.8 ± 0.5% | 77.8 ± 0.4% | 78.2 ± 0.5% | 78.2 ± 0.5% | 78.2 ± 0.5% |
| 19669 | >767 | 41 | 80.2 ± 0.3% | 80.3 ± 0.7% | 78.2 ± 0.5% | 78.2 ± 0.5% | 78.2 ± 0.5% | 78.2 ± 0.5% |
| 19669 | >767 | 82 | 80.3 ± 0.1% | 80.3 ± 0.1% | 78.6 ± 0.3% | 78.6 ± 0.3% | 78.6 ± 0.3% | 78.6 ± 0.3% |

Fig. 2: iCub performing the object acquisition procedure (left) and some examples of the 28 household object classes from the iCubWorld28 dataset (right).

The problem is based on the 10-class MNIST digits classification problem. Our interest lies in studying the accuracy of the classifier when a new class with relatively few examples is included in the model. To this aim, we consider the following experimental procedure for the RLSC models with exact rebalancing, labels recoding and neither of the two:

1) Random choice of a balanced validation set.
2) Pre-training and model selection on 9 classes out of 10, using more n_{bal} training points per class.
3) Training and hold-out cross validation on the remaining class, with various numbers of imbalanced class samples n_{imb}.

This procedure is repeated 20 independent times. In Table II we compare the test accuracies of the three models, when the class “9” is underrepresented with respect to the ones from “0” to “8”. Interestingly, in the imbalanced settings (in which n_{imb} < n_{bal}), the accuracy of the batch model with exact rebalancing is the highest, followed by the incremental model with recoding and the naïve model with neither rebalancing, nor recoding.

B. Robotic Object Recognition Validation on iCub

We evaluated our recursive approach on a dataset designed to benchmark the recognition capabilities of learning systems in realistic robotic scenarios. Specifically, we considered the iCubWorld28 dataset [11] which is characterized by a collection of 28 different objects evenly organized in 7 categories. The dataset has been collected across multiple Human-Robot Interaction sessions during which a human supervisor was showing the objects to the robot from different points of view and lighting conditions. Figure 2 depicts the 28 objects in iCubWorld28, together with an example of the acquisition procedure.

As a visual representation for the images in the dataset in our experiments we used the fc7 layer activations of a CaffeNet [12] pre-trained on the ImageNet large-scale image recognition challenge dataset [13]. The network implements a variant of the model proposed in [14] and is publicly available for download from the Caffe Deep Learning Framework [12].

We repeat the analogous experiment conducted on MNIST to assess classification accuracy in presence of class imbalance during learning. Model selection for λ and α is performed on a separate balanced validation set. In these experiments the best α was found to be 1/2 rather than 1, suggesting that more training data is needed to have a better estimate of class probabilities in this setting.

Accuracy is reported in Table III and confirm that recoding consistently outperforms the competitors, even the rebalanced approach.

Finally, in Figure 3 we report the accuracies of the naïve and recoded incremental models on the entire iCubWorld28 dataset while learning the imbalanced classes 5 (top) and 28 (bottom). On the x-axis, the number of model updates is reported, while on the y-axis the accuracy computed on the test set is shown. It can be noticed that the incremental model with recoding achieves significantly higher accuracies than the naïve incremental ones, especially in the initial, most imbalanced stage. Regarding training time complexity, note in Figure 4 that the proposed incremental model displays substantially equivalent performances to the naïve one, despite the additional labels recoding.

Given these accuracy and time results, we can conclude that the incremental model with recoding attains notable predictive performance improvements, with no additional asymptotic cost with respect to naïve RRLS for classification. Applying the proposed method in the robotic object recognition setting allows for the agent to scale up to larger numbers of samples, still maintaining high accuracy and real-time training computational performances.

CONCLUSIONS

In this paper, we addressed the problem of robotic visual object recognition, analyzing two main issues affecting
Algorithm 1 Recursive regularized least squares for multi-class classification with class extension and recoding.

Input: Training Set \((x_i, y_i)_{i=1}^n\)
- Regularization parameter \(\lambda\)
- Recoding parameter \(\alpha\)

Output: Learned weights \(W_k\) at each step

\[
X_0 \leftarrow [] \\
Y_0 \leftarrow [] \\
R_0 \leftarrow \sqrt{M_d} \\
C \leftarrow \{\emptyset\} \\
\text{for } k = 1 \text{ to } n \text{ do} \\
\quad X_k \leftarrow [X_{k-1}^\top, x_k]^\top \\
\quad \text{if } (y_k \not\in C) \text{ then} \\
\quad\quad C \leftarrow C \cup y_k \\
\quad\quad Y_k \leftarrow \text{addZerosColumn}(Y_{k-1}) \\
\quad\text{end if} \\
\quad Y_k \leftarrow [Y_{k-1}^\top, y_k]^\top \\
\quad \Gamma_k \leftarrow \text{computeGamma}(Y_k) \\
\quad R_k \leftarrow \text{choleskyUpdate}(R_{k-1}, x_k, '+', '+') \\
\quad b_k \leftarrow X_k \Gamma_k^{-1} Y_k \\
\quad W_k \leftarrow R_k^{-1}(R_k^{-\top} b_k) \\
\text{end for}
\]

Fig. 3: Test accuracies of the naïve (red) and recoded (blue) RRLSC models during imbalanced learning of object classes 5 (top) and 28 (bottom).

Fig. 4: Cumulative training times (in seconds) of the naïve (red) and recoded (blue) RRLSC models during incremental learning of the imbalanced object class 28.

Besides, we showed that model selection can be performed via incremental cross validation. In the experimental Section, we confirmed our theoretical analysis with benchmarking experiments and validated the proposed learning algorithm in a real-world humanoid robotics scenario. Remarkably, the experimental results clearly showed the improved predictive performances of our incremental approach, especially in very unbalanced conditions. In addition, the computational requirements at training time have been proven to be equivalent to the recursive RLS ones, thus allowing for real-time implementation and reactive response of the robotic agent to external stimuli.

REFERENCES

[1] S. Thrun and T. M. Mitchell, *Lifelong robot learning*. Springer, 1995.
[2] R. M. Rifkin, “Everything old is new again: a fresh look at historical approaches in machine learning,” Ph.D. dissertation, Massachusetts Institute of Technology, 2002.
[3] R. Rifkin, G. Yeo, and T. Poggio, “Regularized least-squares classification,” Nato Science Series Sub Series III Computer and Systems Sciences, vol. 190, pp. 131–154, 2003.
[4] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. von Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, A. Bernardino, and L. Montesano, “The iCub Humanoid Robot: An Open-systems Platform for Research in Cognitive Development,” Neural Netw., vol. 23, no. 8-9, pp. 1125–1134, Oct. 2010.
[5] I. Steinwart and A. Christmann, *Support vector machines*. Springer Science & Business Media, 2008.
[6] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe, “Convexity, classification, and risk bounds,” Journal of the American Statistical Association, vol. 101, no. 473, pp. 138–156, 2006.
[7] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*. Cambridge university press, 2004.
[8] A. H. Sayed, *Adaptive Filters*. Wiley-IEEE Press, 2008.
[9] G. H. Golub and C. F. V. Loan, *Matrix Computations*, 3rd ed., 1996.
[10] Å. Björck, *Numerical Methods for Least Squares Problems*. Siam Philadelphia, 1996.
[11] G. Pasquale, C. Carlo, F. Odone, L. Rosasco, and L. Natale, “Teaching icub to recognize objects using deep convolutional neural networks,” in *Proceedings of the 4th Workshop on Machine Learning for Interactive Systems*, 2015, p. 21.
[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in *Proceedings of the ACM International Conference on Multimedia - MM ’14*. ACM Press, 11 2014, pp. 675–678.
[13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “Imagenet large scale visual recognition challenge,” p. 43, 9 2014.
[14] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.