CoReS: Compatible Representations via Stationarity

Nicolò Biondi, Federico Pernici, Matteo Bruni, and Alberto Del Bimbo, Senior Member, IEEE

Abstract—Compatible features enable the direct comparison of old and new learned features allowing to use them interchangeably over time. In visual search systems, this eliminates the need to extract new features from the gallery-set when the representation model is upgraded with novel data. This has a big value in real applications as re-indexing the gallery-set can be computationally expensive when the gallery-set is large, or even infeasible due to privacy or other concerns of the application. In this paper, we propose CoReS, a new training procedure to learn representations that are compatible with those previously learned, grounding on the stationarity of the features as provided by fixed classifiers based on polytopes. With this solution, classes are maximally separated in the representation space and maintain their spatial configuration stationary as new classes are added, so that there is no need to learn any mappings between representations nor to impose pairwise training with the previously learned model. We demonstrate that our training procedure largely outperforms the current state of the art and is particularly effective in the case of multiple upgrades of the training-set, which is the typical case in real applications.

Index Terms—Compatible learning, deep convolutional neural network, fixed classifiers, representation learning.

I. INTRODUCTION

Atual intelligent systems learn from visual experience and seamlessly exploit such learned knowledge to identify similar entities. Modern artificial intelligence systems, on their turn, typically require distinct phases to perform such visual search. An internal representation is first learned from a set of images (the training-set) using Deep Convolutional Neural Network models (DCNNs) and then used to index a large corpus of images (the gallery-set). Finally, visual search is obtained by identifying the closest images in the gallery-set to an input query-set by comparing their representations. Successful applications of learning feature representations are: face-recognition, person re-identification, image retrieval, person re-identification, image retrieval, and car re-identification among others.

In the case in which novel data for the training-set and/or more recent or powerful network architectures become available, the representation model may require to be upgraded to improve its search capabilities. In this case, not only the query-set but also all the images in the gallery-set should be re-processed by the upgraded model to generate new features and replace the old ones to benefit from such upgrading. The re-processing of the gallery-set is referred to as re-indexing (Fig. 1).

For visual search systems with a large gallery-set, such as in surveillance systems, social networks or in autonomous robotics, re-indexing is clearly computationally expensive and has critical deployment, especially when the working system requires multiple upgrades or there are real-time constraints. Re-indexing all the images in the gallery-set can be also infeasible when, due to privacy or ethical concerns, the original gallery images cannot be permanently stored and the only viable solution is to continue using the feature vectors previously computed. In all these cases, it should be possible to directly compare the upgraded features of the query with the previously learned features of the gallery, i.e., the new representation should be compatible with the previously learned representation.

Learning compatible representation has recently received increasing attention and novel methods have been proposed in [18], [20], [21], [22], [23], [24]. Differently from these works, in this paper we address compatibility leveraging the stationarity of the learned internal representation. Stationarity allows to maintain the same distribution of the features over time so that it is possible to compare the features of the upgraded representation with the previously learned one.

Fig. 1. Upgrading the DCNN representation model with novel data, typically requires the gallery-set to be re-indexed. Learning compatible representations allows to compare the newly learned representation of an input query-set with the old representation of the gallery-set, thus eliminating its computationally intensive re-indexing.
representation with those previously learned. In particular, we
enforce stationarity by leveraging the properties of a family of
classifiers whose parameters are not subject to learning, namely
\textit{fixed classifiers} based on regular polytopes [25], [26], [27], that
allow to reserve regions of the representation space to future
classes while classes already learned remain in the same spatial
configuration.

The main contributions of our research are the following:

1) We identify stationarity as a key property for compatibility
and propose a novel training procedure for learning com-
patible feature representations via stationarity, without the
need of learning any mappings between representations
nor to impose pairwise training with the previously learned
model. We called our method: Compatible Representa-
tions via Stationarity (CoReS).

2) We introduce new criteria for comparing and evaluating
compatible representations in the case of sequential multi-
model upgrading.

3) We demonstrate through extensive evaluation on large
scale verification, re-identification and retrieval benchmarks
that CoReS improves the current state-of-the-art in
learning compatible features for both single and sequential
multi-model upgrading.

In the following, in Section II, we discuss the main literature
on compatible representation learning and highlight the distin-
guishing features of our solution. In Section III, we present
in detail the problem of learning compatible representations
and define new criteria and metrics for compatibility evalua-
tion in sequential multi-model upgrading. In Section IV, we
present our solution for learning compatible representations by
exploiting feature stationarity. In Section V, we evaluate our
solution against state-of-the-art methods on different benchmark
datasets and network architectures and demonstrate its superior
performance in learning compatible representations. Finally, in
Section VI, we perform an extensive ablation study.

II. RELATED WORKS

\textbf{Compatible Representation Learning:} The term \textit{backward}
compatibility was first introduced in [28] for the classification
task. They noted that although machine learning models can
increase on average their performance with the availability of
more training data, upgrading the model could result into incor-
correct classification of data correctly classified with the previous
model. As a consequence, the trust in machine learning systems
is severely harmed. Compatibility in classification has been
further investigated in [29], [30], [31], [32].

However, learning compatible representations is substantially
different from learning compatible classifier models, although
both follow the same general principle. As a distinct hallmark,
learning compatible representations directly imposes constraints
in the semantic distance of the feature representation. In [21],
[22], [23], [24] the problem of feature compatibility was ad-
dressed by learning a \textit{mapping} between two representation
models so that the new and old feature vectors can be directly
compared. The mapping in [21] was learned through a three-
step procedure: adversarial learning for reconstruction, feature
extraction and regression to jointly optimize the whole model.

In [22], the mapping was learned through an autoencoder by
minimizing the distance between the two representation spaces
and the reconstruction error. In [23], the mapping was learned
from a residual bottleneck transformation module trained by
three different losses: classification loss, similarity loss between
feature spaces, and KL-divergence loss between the prototypes
of the classifiers. In [24], the estimated mapping aligns the
class prototypes between the models. To further encourage
compatibility, the method also reduces intra-class variations for
the new model. All these methods do not completely avoid the
cost of re-indexing of the gallery-set as, at each upgrade, the
old feature vectors must be re-processed with the learned map-
ping. Therefore, they are not suited for sequential multi-model
learning and large gallery-sets. Differently from these works,
we avoid learning specific space-to-space mappings for each
previous upgraded representation model and completely avoid
the cost of re-indexing also in the case of multiple upgrades.

By avoiding to learn space-to-space mappings, our work has
some affinity with the Backward Compatible Training
(BCT) [18] method for compatible learning that represents the
current state-of-the-art. BCT grounds on pairwise compatibility
learning to obtain compatible features. It takes advantage of an
influence loss that biases the new representation in a way that
it can be used by both the new and the old classifier. During
learning with novel data, the old classifier is \textit{fixed} and the pro-
totypes of the new classifier align with the prototypes of the old
classifier. In the case of multi-model upgrading, such pairwise
cooperation supports compatibility only indirectly (i.e., through
transitive compatibility). In fact, for a two-model upgrading,
i.e., when the model \( \phi_1 \) is upgraded to \( \phi_2 \) and \( \phi_2 \) to \( \phi_3 \), \( \phi_3 \) is
compatible with \( \phi_1 \) thanks to the compatibility of \( \phi_3 \) with \( \phi_2 \)
and of \( \phi_2 \) with \( \phi_1 \). BCT has been extended in [20], where small
and large representation models are taken into account for the
query and the gallery-set, respectively. Differently from BCT our
method is not based on pairwise compatibility learning and does
not use previous classifiers which might be incorrectly learned.
Instead, we learn a representation that is directly compatible to
\textit{all} the previous representations by following a training strategy
that only leverages feature stationarity.

A different compatible representation learning scenario, re-
ferred to as asymmetric metric learning, was addressed in [33].
Large network architectures are used for the gallery-set and
smaller architectures for queries without considering new data
for the training-set.

Compatibility was implicitly studied also in [34], [35] in
which representation similarity between two networks with
identical architecture but trained from different initializations
was evaluated.

\textbf{Neural Collapse:} Our method is based on the concept
of learning stationary and maximally separated features using
the \textit{d-Simplex fixed} classifier introduced in [26] and [27]. In this
classifier, weights are not trainable and are determined from the
coordinates of the vertices of the \textit{d-Simplex} regular polytope.
The goal of learning stationary and maximally separated features
has similarities to the Neural Collapse phenomenon described
in [36]. This phenomenon, which can be further explored in [37],
shows that the final learnable classifier and the learned feature representation tend to orient towards a $d$-Simplex structure. Along this line of research, the works [38] and [39] investigated fixing the final classifier in the scenario of balanced datasets. In contrast, in [40] it is shown that training on imbalanced datasets does not lead to Neural Collapse. However, [41] shows that Neural Collapse can occur in imbalanced and long-tail scenarios as long as the classifier is fixed to a $d$-Simplex, and this evidence has been recently supported in [42] also for out-of-distribution detection. Our experimental results on large long-tailed datasets with distribution shifts, such as [43] and [44], demonstrate that learning stationary and maximally separated features is effective even in scenarios with increased complexity due to the extra challenge of finding compatible representations.

Class-Incremental Learning: Class-incremental Learning (CiL) is the process of sequentially increasing the number of classes to learn over time [45], [46], [47]. Although apparently similar to sequential learning of compatible representations, the main focus of CiL is to reduce catastrophic forgetting [48]. CiL differs from compatible representations learning in two important aspects: (1) in CiL the new model is typically initialized with the old model, while in compatible representation learning this is optional (2) in CiL the new model has not access to the whole data during the model upgrade, while in compatible representation learning all training data is available at each upgrade step. According to this, in compatible representation learning, the learned representation is not affected by catastrophic forgetting.

III. Compatibility Evaluation

We indicate with $I_G = \{x_i\}_{i=1}^N$ and $F_G = \{f_i\}_{i=1}^N$ respectively the set of images and their features in the gallery-set $G$. The gallery-set $G$ might be grouped into a number of classes or identities $L$ according to a set of labels $Y = \{y_i\}_{i=1}^L$. We assume that the features $F_G$ are extracted using the representation model $\phi_{old} : \mathbb{R}^D \to \mathbb{R}^d$ that transforms each image $x \in \mathbb{R}^D$ into a feature vector $f \in \mathbb{R}^d$, where $d$ and $D$ are the dimensions of the feature and the image space, respectively. Analogously, we will refer to $I_Q$ and $F_Q$ respectively as the set of images and their features in the query-set $Q$. The model $\phi_{old}$ is trained on a training-set $T_{old}$ and used to perform search tasks using a distance $\text{dist} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ to identify the closest images to the query images $I_Q$. As novel images $X$ become available, a new training-set $T_{new} = T_{old} \cup X$ is created and exploited to learn a new model $\phi_{new} : \mathbb{R}^D \to \mathbb{R}^d$ that improves (i.e., upgrades) the $\phi_{old}$ model. Our goal is to design a training procedure to learn a compatible model $\phi_{new}$ so that any query image transformed with it can be used to perform search tasks against the gallery-set directly without re-indexing, i.e., without computing $F_G = \{f \in \mathbb{R}^d | f = \phi_{new}(x) \forall x \in I_G\}$.

A. Compatibility Criterion

In [18], a general criterion to evaluate compatibility was defined. According to this, a new and compatible representation model must be at least as good as its previous version in clustering images from the same class and separating those from different classes. A new representation model $\phi_{new}$ is therefore compatible with an old representation model $\phi_{old}$ if

$$\text{dist}(\phi_{new}(x_u), \phi_{old}(x_v)) \leq \text{dist}(\phi_{old}(x_u), \phi_{old}(x_v))$$

$$\forall (u,v) \in \{(u,v) | y_u = y_v\}$$

and

$$\text{dist}(\phi_{new}(x_u), \phi_{old}(x_v)) \geq \text{dist}(\phi_{old}(x_u), \phi_{old}(x_v))$$

$$\forall (u,v) \in \{(u,v) | y_u \neq y_v\},$$

where $x_u$ and $x_v$ are two input samples and $\text{dist}(\cdot,\cdot)$ is a distance in feature space. Since it constrains all pairs of samples, (1) is relaxed to the following Empirical Compatibility Criterion:

$$M(\phi_{new}, \phi_{old}) > M(\phi_{old}, \phi_{old}),$$

where $M$ is a metric used to evaluate the performance based on $\text{dist}(\cdot,\cdot)$. The notation $M(\phi_{new}, \phi_{old})$ underlines that the upgraded model $\phi_{new}$ is used to extract feature vectors $F_G$ from query images $I_Q$, while the old model $\phi_{old}$ is used to extract features $F_G$ from gallery images $I_G$. This performance value is referred to as cross-test. Correspondingly, $M(\phi_{old}, \phi_{old})$ evaluates the case in which both query and gallery features are extracted with $\phi_{old}$ and is referred to as self-test. The underlying intuition of (2) is that model $\phi_{new}$ is compatible with $\phi_{old}$ when the cross-test is greater than the self-test, i.e., by using the upgraded representation for the query-set and the old representation for the gallery-set the system improves its performance with respect to the previous condition.

To evaluate the relative improvement gained by a new learned compatible representation, the following Update Gain has been defined:

$$\Gamma(\phi_{new}, \phi_{old}) = \frac{M(\phi_{new}, \phi_{old}) - M(\phi_{old}, \phi_{old})}{M(\phi_{new}, \phi_{new}) - M(\phi_{old}, \phi_{old})},$$

where $M(\phi_{new}, \phi_{new})$ stands for the best accuracy level we can achieve by re-indexing the gallery-set with the new representation [18] and can be considered as the upper bound of the best achievable performance.

B. Multi-Step Compatibility Criterion

In real world applications, multi-step upgrading is often required, i.e., different representation models must be sequentially learned through time, in multiple upgrade steps. At each step $t$, the training-set is upgraded as

$$T_t = T_{t-1} \cup X_t,$$

being $X_t$ the new data and $T_{t-1}$ the training-set at step $t - 1$. In the multi-step upgrading case, we define the following Multi-model Empirical Compatibility Criterion as follows:

$$M(\phi_{t'}, \phi_{t}) > M(\phi_{t'}, \phi_{t'}) \quad \forall t' > t$$

with $t' \in \{2, 3, \ldots, T\}$ and $t \in \{1, 2, \ldots, T - 1\}$, (5) where $\phi_{t'}$ and $\phi_{t}$ are two different models such that $\phi_{t}$ is upgraded before $\phi_{t'}$, $T$ is the number of upgrade steps and $M$ the metric used to evaluate the performance. Model $\phi_{t}$ is compatible with $\phi_{t'}$ when their cross-test is greater than the self-test of $\phi_{t}$.
for each pair of upgrade steps. Fig. 2 illustrates the Multi-model Empirical Compatibility Criterion, where \( \{ \phi_1, \phi_2, \ldots, \phi_T \} \) are the representation models, black arrows indicate the model upgrades and gray arrows represent self and cross-tests.

In order to assess multi-model compatibility of (5) for a sequence of \( T \) upgrade steps, we define the following square triangular Compatibility Matrix \( C \):

\[
C = \begin{bmatrix}
M (\phi_1^\circ, \phi_1^\circ) & M (\phi_2^\circ, \phi_1^\circ) & M (\phi_3^\circ, \phi_2^\circ) & \cdots & M (\phi_T^\circ, \phi_2^\circ) \\
M (\phi_2^\circ, \phi_1^\circ) & M (\phi_2^\circ, \phi_2^\circ) & \cdots & \cdots & M (\phi_T^\circ, \phi_3^\circ) \\
M (\phi_3^\circ, \phi_2^\circ) & \cdots & \cdots \cdots & \cdots & M (\phi_T^\circ, \phi_T^\circ) \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
M (\phi_T^\circ, \phi_T^\circ) & \cdots & \cdots & \cdots & M (\phi_T^\circ, \phi_T^\circ)
\end{bmatrix},
\]

where each entry \( C_{ij} \) is the performance value according to metric \( M \), taking model \( \phi_i \) for the query-set \( Q \) and model \( \phi_j \) for the gallery-set \( G \). Entries on the main diagonal, \( i = j \), represent the self-tests, while the entries off-diagonal, \( i > j \), represent the cross-tests. While showing compatibility performance across multiple upgrade steps, matrix \( C \) can be used to provide a scalar metric to quantify the global multi-model compatibility in a sequence of upgrade steps. In particular, we define the Average Multi-model Compatibility (AC) as the number of times that (5) is verified with respect to all its possible occurrences, independently of the number of the learning steps

\[
AC = \frac{2}{T(T-1)} \sum_{i=2}^{T} \sum_{j=1}^{i-1} \mathbb{I} (C_{ij} > C_{jj}),
\]

where \( \mathbb{I}(\cdot) \) denotes the indicator function.

Finally, we define the Average Multi-model Accuracy (AM) as the average of the entries of the Compatibility Matrix

\[
AM = \frac{2}{T(T+1)} \sum_{i=1}^{T} \sum_{j=1}^{i} C_{ij},
\]

to provide an aggregate value of the accuracy metric \( M \) under compatible training.

IV. LEARNING COMPATIBLE REPRESENTATIONS

It is well known that for different initializations a neural network learns the same subspaces but with different basis vectors [34], [35]. Therefore, training the network from scratch with different randomly initialized weights does not provide similar representations in terms of subspace geometry. This result excludes compatibility between two independently trained representation models.

The alternative of learning with incremental fine-tuning (i.e., weights are initialized from the previously learned model) appears to be a more favorable training procedure to compatibility. However, and perhaps counterintuitively, this does not help to keep the same subspace representation geometry regardless of the changes made.

We provide direct evidence of this aspect of feature learning in Fig. 3 with a toy problem. We trained the LeNet++ architecture [49] on a subset of the MNIST dataset setting the output size of the last hidden layer to two (so resulting in a two-dimensional representation space). The classifier weights were unit normalized and biases were set to zero to encourage learning cosine distance between features ([50], [51]) and the cross-entropy loss was used. The model was initially trained with five classes (red, orange, blue, purple, and green clouds in Fig. 3(a)); then a new class (brown cloud in Fig. 3(b)) was included in the training-set and the new model was trained by fine-tuning the old model on the new training-set of six classes. As the new class is included in the training-set and the representation is fine-tuned, the features of the old classes change their spatial configuration and the mutual angles between classifier prototypes change as well. This is due to the fact that linear classifiers maximize inter-class distance to better discriminate between classes [49]. As a consequence, the cosine distance comparison between old and new features cannot be guaranteed. The same effect holds for any number of classes and feature space dimension.

To limit such spatial configuration changes and therefore achieve feature compatibility, our approach learns stationary features exploiting the properties of fixed classifiers introduced in [27] that we briefly recall in the next subsection.

A. Learning Stationary Features With Fixed Classifiers

In [52], [53], a DCNN model with a fixed classification layer (i.e., not subject to learning) initialized by random weights was
shown to be almost equally effective as a trainable classifier with substantial saving of computational and memory requirements. In fixed classifiers, the functional complexity of the classifier is fully demanded to the internal layers of the neural network. As the parameters of the classifier prototypes are not trainable, only the feature vector directions align toward the directions of the prototypes. In [27], we presented a special class of fixed classifiers where the weights of the classifier are fixed to values taken from the coordinates of the vertices of regular polytopes. Regular polytopes generalize regular polygons in any number of dimensions, and reflect the tendency of splitting the available space into approximately equiangular regions. There are only three possible polytopes in a multi-dimensional feature space with dimensions 5 and higher: the d-Simplex, the d-Cube and the d-Orthoplex. In these classifiers, the spatial configuration of the learned classes remains stationary as new classes are added and, at the same time, classes are maximally separated in the representation space [25].

### B. Learning Compatibility via Stationarity

Compatibility has a close relationship with feature stationarity. In fact, stationarity requires that the representation that is learned in the future is statistically indistinguishable from the representation learned in the past, regardless of the number of model upgrades performed over time. We therefore argued that a compatibility training procedure should be defined by directly exploiting stationarity of the representation as provided by fixed classifiers.

According to this, in order to maintain the compatibility across model upgrades, at each upgrade we learn the features of the new classes in reserved regions of the representation space, while the features learned in the previous upgrades will not “move” thanks to the stationarity property. Region reservation is made at the beginning of training by setting a number of classifier outputs greater than the number of the initial classes and keeping not assigned regions for future upgrades. Fig. 4 provides an overview of the CoReS procedure for compatible representation learning. Fig. 4(a) shows the initial training-set \( T_{\text{old}} \) (two classes for simplicity), the new data \( \mathcal{X} \) (two classes for simplicity), the regions left for the future classes, the class prototypes \( \mathbf{w}_i \), and the softmax outputs of the fixed classifier. As a result, the training-set \( T_{\text{new}} \) that will be used to upgrade the initial representation has four classes. Fig. 4(c) shows the representation space of a 10-sided 2D regular polytope (a polygon) with the old and new classes, their prototypes and class samples. The regions reserved for the future classes are colored in gray. Fig. 4(b) highlights the coordinates of one of the fixed classifier prototypes.

As new data \( \mathcal{X}_t \) is available at time \( t \) to upgrade the representation, CoReS performs optimization of model \( \phi_t \) according to the following loss:

\[
L_t = -\frac{1}{N_t} \sum_{i=1}^{N_t} \log \left( \sum_{j=1}^{K} e^{\mathbf{W}^\top_{n_i} \phi_t(x_i) + \sum_{j=K_t+1}^{K} e^{\mathbf{W}^\top_j \phi_t(x_i)}} \right),
\]

where: \( x_i \in \mathcal{T}_{t-1} \cup \mathcal{X}_t \) is a sample instance; \( y_i \in \{1, 2, \ldots, K_t\} \) its class; \( \phi_t(x_i) \in \mathbb{R}^d \) its feature vector as from model \( \phi_t \); \( \mathbf{W} \in \mathbb{R}^{d \times K} \) the fixed classifier weight matrix; \( K_t \) and \( K \), respectively the number of classes in \( \mathcal{T}_{t-1} \cup \mathcal{X}_t \) and the number of classifier outputs; and \( N_t \) the number of samples in the mini-batch.

For the \( K - K_t \) outputs that have not yet been associated to classes at time \( t \), the classifier responds with false positives of the training-set \( T_t \). To highlight the role of these outputs, in the denominator of the loss in (9), we distinguished the contributions corresponding to the \( K_t \) classes at time \( t \) (the first summation) from the \( K - K_t \) future classes (the second summation). The contribution of the future classes enforces an angular margin penalty in the representation space that tends to push away the features already learned from the prototypes of future classes. Such a margin penalty has a key role for learning compatible features as, when upgrading the model with new data, it prevents...
the new data to affect the representation already learned. As is evident from (9), CoReS learns feature compatibility without using the previously learned model or classifier.

In our implementation, CoReS uses the $d$-Simplex fixed classifier as it exploits the only polytope that provides class prototypes equidistant from each other [27]. In this way, we avoid bias in the assignment of class labels to classifier outputs. The $d$-Simplex fixed classifier defined in a feature space of dimension $d$ can accommodate a number of classes equal to its number of vertices, i.e., $K = d + 1$. As a result, the $K$ classifier prototypes are computed as

$$W = \left[ e_1, e_2, \ldots, e_{K-1}, \frac{1 - \sqrt{K}}{K - 1} \sum_{i=1}^{K-1} e_i \right],$$

where $e_i$ denotes the standard basis in $\mathbb{R}^d$, with $i \in \{1, 2, \ldots, K-1\}$. Increasing the dimensionality of the representation has no effect on both the training time and memory consumption, since fixed classifiers do not require back-propagation on the fully connected layers [27], [53].

### C. CoReS Extension by Model Selection

At each upgrade, the training-set grows. This causes a shift in the distribution between the training-set and the gallery-set [54]. To cope with this shift, we extended CoReS with a simple model selection strategy, based on cross validation.

We assume that: (1) drifting occurs gradually such that the density ratio of the marginal distribution before and after the upgrade is close to uniform [55]; (2) there is a set of data $\mathcal{G}$ available that is an i.i.d. sample of the gallery-set data $\mathcal{G}$ (i.e., a split following the same distribution); (3) the previously learned model is available. Under these assumptions, at each upgrade step, for each epoch, we search the model with the highest self-test among the models that satisfy the compatibility criterion of (2) with respect to the previous model

$$\phi_t = \arg \max_l M \left( \phi_{t,l}, \phi_{t,d}^{\mathcal{G}} \right) \quad \text{s. t.} \quad M \left( \phi_{t,l}, \phi_{t,d}^{\mathcal{G}} \right) > M \left( \phi_{t}, \phi_{t,d}^{\mathcal{G}} \right),$$

where $t = t' - 1$ and $l$ is the index of the training epoch.

### V. EXPERIMENTAL RESULTS

In this section, we compare CoReS against baselines and the Backward Compatible Training (BCT) [18] state-of-the-art method for different visual search tasks on different benchmark datasets. In particular, in Section V-A we evaluate open-set verification with single and multi-model upgrading on the Cifar-100/10 datasets [57]; in Sections V-B and V-C, we analyze single and multi-model upgrading in more challenging tasks, namely face verification (in Section V-B) on the CASIA-WebFace/LFW datasets [58], [59] and person re-identification (in Section V-C) on the Market1501 dataset [60]; in Sections V-D and V-E, we evaluate CoReS in challenging long-tail distribution datasets, namely Google Landmark Dataset v2 [43] and MET dataset [44]. The original datasets are split so that training-sets used for each upgrade have the same number of classes. Finally, in Section V-F, we report a qualitative analysis that provides evidence of how stationarity contributes to compatibility. All the representation models were trained using 4 NVIDIA Tesla A-100 GPUs.

#### A. Compatibility Evaluation on Cifar-100/10

We perform open-set verification with Cifar-100/10 datasets. The Cifar-100 and Cifar-10 datasets consist of 60,000 $32 \times 32$ RGB images (50,000 for training and 10,000 for test) in 100 and 10 classes, respectively. Their classes are mutually exclusive. We use the Cifar-100 training-set to create the training-sets for upgrading the models and the Cifar-10 test-set for generating the verification pairs used in the open-set verification protocol. We do not use the test-set of Cifar-100 and the training-set of Cifar-10.

The SENet-18 architecture [61] adapted to the Cifar $32 \times 32$ image size is used. The same random seed is used for initialization every time the model is upgraded. For CoReS, the output nodes of the $d$-Simplex fixed classifier are set to $K_p = 100$ so resulting in a feature space is 99-dimensional. Optimization is performed using SGD with 0.1 learning rate, 0.9 momentum, and $5 \times 10^{-4}$ weight decay. The batch size is set to 128. With every upgrade, training is terminated after 100 epochs. Learning rate is scheduled to decrease to 0.01 at 70 epochs.

We compared CoReS against the $\ell^2$, the Incremental Fine-Tuning (IFT), the Learning without Forgetting (LwF) [56] baselines and the Backward Compatible Training (BCT) method [18]. All the baselines except IFT, are the public implementations of [18]. The $\ell^2$ approach achieves compatibility using an auxiliary loss that takes into account the euclidean distance $\ell^2$ between features of images in $\mathcal{T}_{old}$ as evaluated by $\phi_{new,\ell^2}$ and $\phi_{old}$, that is

$$L_{dist} = \frac{1}{|\mathcal{T}_{old}|} \sum_{x \in \mathcal{T}_{old}} \text{dist}(\phi_{new,\ell^2}(x), \phi_{old}(x)).$$

According to this, the new representation model $\phi_{new,\ell^2}$ is trained as

$$\phi_{new,\ell^2} = \arg \min_{\phi} \left( \mathcal{L}(\phi, \mathcal{T}_{new}) + \lambda L_{dist}(\phi, \phi_{old}, \mathcal{T}_{old}) \right),$$

where $\mathcal{L}$ is the standard cross-entropy loss and the scalar $\lambda$ balances the two losses. The baseline ITM (Independently Trained Models) learns the representation of two versions of models trained independently with standard cross-entropy loss. When the gallery is re-indexed, ITM may be thought of as the standard approach to observe, if any, a reduction in performance of a compatibility-based method. The decrease is meant to accommodate the increased requirement that the learning model must fulfill to learn a compatible representation. ITM and CoReS are related since both methods use standard cross-entropy loss and linear classifiers. This relationship allows to quantify the representation capability used for compatibility purposes. Indeed,

1.SENet-18 code: https://github.com/kuangliu/pytorch-cifar
2.https://github.com/YantaoShen/openBCT

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
TABLE I

| COMPARISON PAIR | M | ECC | 𝛾 (%) | ABSOLUTE GAIN |
|-----------------|---|-----|--------|---------------|
| (φ_{0|3}, φ_{1|3}) (Lower Bound) | 0.60 | − | − | − |
| (φ_{0|10−ITM}, φ_{1|3}) | 0.12 | × | − | − |
| (φ_{0|10−ITM}, φ_{1|4}) | 0.35 | × | − | − |
| (φ_{0|10−LwF}, φ_{1|3}) | 0.57 | × | − | − |
| (φ_{0|10−CoReS}, φ_{1|3}) | 0.59 | × | − | − |
| (φ_{0|10−CoReS}, φ_{1|4}) | 0.60 | √ | 5.9 | 0.20 |

TABLE II

| COMPARISON PAIR | CoReS | BCT |
|-----------------|-------|-----|
| M | ECC | 𝛾 (%) | M | ECC | 𝛾 (%) |
| (φ₁, φ₁) | 0.59 | − | 0.59 | − |
| (φ₂, φ₁) | 0.61 | √ | 54.7 | 0.59 | √ | 5.2 |
| (φ₃, φ₁) | 0.60 | √ | 18.5 | 0.58 | × | − |
| (φ₂, φ₂) | 0.63 | − | − | 0.61 | − | − |
| (φ₃, φ₂) | 0.61 | × | − | 0.59 | × | − |
| (φ₃, φ₃) | 0.65 | − | − | 0.64 | − | − |

since the self-test values in the compatibility matrices are calculated with both the query and the gallery images extracted from the same model (i.e., re-indexing the gallery), the difference of the main diagonal entries of ITM and CoReS quantifies the neural network model capacity used to learn compatibility.

To perform open-set verification, we use 3,000 positive and 3,000 negative pairs randomly generated from the Cifar-10 test-set. Given a pair of images of Cifar-10, verification assesses whether they are of the same class using the cosine distance. One of the images of each pair can be considered gallery and the other the query. Compatibility is evaluated for one, two, four and nine upgrade steps.

In the one-upgrade case, the model is trained with 50% of Cifar-100 classes and upgraded using 100%. The comparative evaluation is shown in Table I in terms of verification accuracy (M), whether the Empirical Compatibility Criterion is verified (ECC) and Update Gain (Γ). It can be noticed that only BCT and CoReS satisfy (2). However, CoReS obtains higher Update Gain with respect to BCT.

In the two-upgrade case, the model is initially trained on 33% of Cifar-100 classes and upgraded with 66% and 100%. The compatibility matrix is shown in Fig. 5 for all the methods compared. We can notice that none of the methods achieves full compatibility in this case, however BCT and CoReS achieve higher accuracy values with respect to the others. CoReS misses compatibility only once between $φ_3$ and $φ_2$. In Table II the behavior of CoReS and BCT in the two-upgrade case is reported in more detail. We can observe that CoReS has a substantially higher compatibility and update gain than BCT. This may be due to the fact that BCT learns compatibility of $φ_3$ with no direct relationship with $φ_1$, while CoReS learns $φ_3$ taking into account both $φ_1$ and $φ_2$, because all models share the same class prototypes in the feature space.

The compatibility matrices of CoReS and BCT for the cases of four and nine upgrade steps are shown in Figs. 6 and 7, respectively. The superior behavior of CoReS with respect to BCT is clearly evident in both cases, scoring 0.9 versus 0.5 and
Fig. 7. Compatibility matrices of CoReS and BCT for open-set verification on CIFAR 100/10 with 9-step multi-model upgrading. Diagonal and off-diagonal elements report self-test and cross-test accuracy, respectively. Models are sequentially learned with 10%, 20%, ..., 100% of training data. Models that do not satisfy compatibility are highlighted in red.

Fig. 8. Compatibility of CoReS and compared methods (shown color-coded) for open-set face verification on the CASIA-WebFace/LFW dataset with multi-model upgrading. Bins show: (a) AC scores for different number of upgrades; (b) AM scores for different number of upgrades.

TABLE III

| COMPARENSION PAIR | CoReS | BCT |
|-------------------|-------|-----|
| \(\phi_{old}, \phi_{old}\) | M | ECC | \(\Gamma(\%)\) | M | ECC | \(\Gamma(\%)\) |
| \(\phi_{new}, \phi_{old}\) | 0.90 | - | - | 0.91 | - | - |
| \(\phi_{new}, \phi_{new}\) | 0.91 | \(\checkmark\) | 0.32 | 0.91 | \(\checkmark\) | 0.29 |

0.51 versus 0.11 of AC, respectively. This indicates that the more upgrades are made, the better CoReS performs with respect to BCT.

B. Compatibility Evaluation on CASIA-WebFace/LFW

In this evaluation, we perform open-set face verification using CASIA-WebFace dataset to create the training-sets and LFW as the test set. The CASIA-WebFace dataset includes 494,414 RGB face images of 10,575 subjects. The LFW dataset contains 13,233 target face images of 5,749 subjects. Of these, 1,680 have two or more images, while the remaining 4,069 have only one single image. ResNet50 [62] with input size of 112 \(\times\) 112 is used as backbone. Optimization is performed using SGD with 0.1 learning rate, 0.9 momentum, and \(5 \cdot 10^{-4}\) weight decay. The batch size is 1,024. With every upgrade, training is terminated after 120 epochs. Learning rate is scheduled to decrease to 0.01, 0.001, and 0.0001 at epoch 30, 60, 90 respectively.

Compatibility is evaluated for one, two, three, four, five, and nine upgrade steps.

In the one-upgrade case, models are learned with 50% of the CASIA-WebFace dataset and upgraded with 100%. Table III shows that, with 50% of CASIA-WebFace dataset, CoReS and BCT achieve similar performance. This is because there is already sufficient data variability to learn compatible features.

A big difference between CoReS and compared methods becomes evident when multiple upgrades are considered as shown in Fig. 8(a). In this figure, the values of the AC are reported for each method over a set of different experiment respectively with one, two, three, four, five, and nine upgrades. CoReS achieves full compatibility (AC = 1) for one, two, three, and four upgrades and starts decreasing AC from five upgrade...
steps up to $AC = 0.58$ with nine upgrades. In contrast BCT loses performance already with two upgrade steps finishing at $AC = 0.09$ with nine upgrade steps. Baseline methods report $AC = 0$ in all of the scenario. The compatibility matrices for the case of four upgrades are shown in detail in Fig. 9(f). In this case, we can appreciate full compatibility of CoReS with respect to the poor compatibility of BCT and other methods. In Fig. 8(b), we report the $AM$ metric for these experiments. It can be observed that the CoReS and BCT score almost the same average verification accuracy in all the experiments, while in the others values are always lower. This is due to the fact that cross-test values are low since no compatibility is reported.

We conclude that for multi-model upgrading, CoReS, while having the same verification performance of BCT, largely improves compatibility across model upgrades with 544% relative improvement over BCT for the challenging scenario of nine-step upgrading. The lower $AC$ of BCT appears to be related to the fact that in this method compatibility is obtained only through transitivity from the model previously learned.

C. Compatibility Evaluation on Market1501

In this experiment, we perform person re-identification (1:N search) using the Market1501 dataset. Differently from CASIA-WebFace/LFW, Market1501 includes images of identities with severe occlusions and pose variations. This makes learning compatible features largely more challenging. The Market1501 dataset contains 1,501 identities, split in 751 for training and 750 for test. The two sets do not share identities. Images of each identity are captured by several cameras so that cross-camera search can be performed. For 1:N re-identification, a set of templates is used in the gallery-set. Then templates of the query-set are used to search against the gallery templates. Mean Average Precision (mAP) is used as quality metric. Following [63], we use a pre-trained ResNet101 [62] as backbone and Adam [64] as optimizer with an initial learning rate of $3 \cdot 10^{-4}$. The batch size is 256. With every upgrade, training is terminated after 25 epochs. Learning rate is scheduled to decrease to $3 \cdot 10^{-5}$ after 20 epochs.

Table IV shows the results for one and two upgrade steps. We observe that CoReS always achieves full compatibility, while BCT loses compatibility already with two upgrade steps. The baseline methods never achieve compatibility in both one and two-step upgrade. While it is evident that the stationarity of the CoReS representation still plays a key role to ensure compatibility across the upgrades also in this challenging task, the results confirm the limits of transitive compatibility used by BCT.

| METHOD | One-upgrade | Two-upgrades |
|--------|-------------|--------------|
|        | $AC$        | $AM$         | $AC$        | $AM$         |
| STANDARD | 0.30        | 0.22         | 0.30        | 0.31         |
| $\ell^2$  | 0.36        | 0.31         | 0.36        | 0.31         |
| IFT       | 0.40        | 0.36         | 0.50        | 0.50         |
| BCT       | 0.49        | 0.50         | 0.50        | 0.50         |
| CoReS     | 0.57        | 0.51         | 0.57        | 0.51         |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
D. Compatibility Evaluation on GLDv2

Google Landmark Dataset v2 (GLDv2), cleaner version, consists of 81,313 classes and 1,580,470 images [43]. The main issue is that with such a large number of classes, the \(d\)-Simplex classifier matrix will be large and will require additional GPU memory. Although we have already evaluated the fixed classifier by training with more than 10 k classes using CASIA-WebFace dataset, with 81 k classes it is necessary to also reduce the representation dimension. This is a result of the linear growth of the feature dimension of the \(d\)-Simplex fixed classifier with the number of classes. However, training can still be achieved effectively because fixed weights do not require additional storage for the gradient, thus resulting in a decrease in the amount of memory needed. The challenging situation remains at test time in which the gallery-set is composed of more than 760 k elements, each of which is 81,312 dimensional, requiring more than 200 Gb of storage. With such a memory requirement, the Faiss GPU accelerated indexing [65] available in the GLDv2 repository\(^3\) may not be fully suitable for use. To overcome this issue, inspired by [66], we adopted top-\(k\) sparsification\(^4\) and perform nearest neighbor using the PySparNN library.\(^5\) Table V shows the retrieval results on GLDv2 with one-step upgrading for both CoRes and BCT. In our experiments, we set the top-\(k\) sparsification to \(k = 512\), use the ResNet50 backbone and image size \(256 \times 256\). As it can be seen from the table, CoRes achieves compatibility also under this learning settings, while BCT results not compatible as the cross-test is 0.16 which is lower than the compatibility also under this learning settings, while BCT results not compatible as the cross-test is 0.16 which is lower than the compatibility. As evidenced by the table, only CoRes achieves compatibility. This finding further supports the positive performance of the proposed method, and suggests that it may also be used for contrastive/self-supervised learning. The positive results may be motivated by the fact that the extension of CoRes to contrastive learning is related to DirectCLR [67] in which the SimCLR contrastive method [68] is trained with a final fixed projector based on random weights. We can consider the \(d\)-Simplex classifier in this “contrastive extended CoRes” as a special fixed projector. The positive results we achieved are consistent with the improved results on SimCLR reported by DirectCLR.

E. Compatibility Evaluation on MET

MET, mini version, is an instance-level artwork recognition dataset consisting of 38,307 images and 33,501 classes [44]. The main issue for this dataset is that standard supervised learning with cross-entropy is not converging. The official repository\(^6\) provides a self-supervised training baseline based on contrastive learning [1] which, starting from a pre-trained model on ImageNet, converges to a working representation. We have extended CoRes to support the contrastive learning setup. The extension consists in using the output logits of the \(d\)-Simplex fixed classifier as the contrasted representation. Since classes are not required in contrastive learning, under this condition the dimension \(d\) of the fixed classifier becomes a free parameter (we set \(d = 2048\)). Our training in miniMET starts by pre-training a \(d\)-Simplex fixed classifier based ResNet50 on ImageNet as in [27]. Contrastive learning with CoRes is then accomplished by using a distinct \(d\)-Simplex fixed final matrix which operates as a projector rather than a classifier. MiniMET is used to create the training-sets for the upgrade and the gallery-set, while the MET dataset of the queries is used as query-set. Table VI shows the result with one-step upgrading of CoRes and BCT.

F. Qualitative Results

Fig. 10 shows the different behaviors of IFT, BCT, and CoRes for a simple two-step upgrading with the MNIST dataset. In this experiment, the training set is upgraded from seven to eight classes and then to nine classes using the same setting as in Section IV.

It can be observed that IFT changes the spatial configuration of the representation as every new class is learned. In fact, IFT has no mechanism to prevent the rearrangement of the features when the model is upgraded. By exploiting the class prototypes of the classifier at the previous upgrade step, BCT keeps the representation reasonably stationary, although small changes are introduced to accommodate the new classes. Differently from the others, the features learned by CoRes remain aligned with the class prototypes. The configuration of the feature space

\(^3\)https://github.com/cvfdoundation/google-landmark
\(^4\)All the entries of a feature representation vector are set to zero except for the top \(k\).
\(^5\)https://github.com/facebookresearch/pysparnn
\(^6\)https://github.com/nikosips/met

### Table V

| Comparison Pair | CoRes | BCT |
|-----------------|-------|-----|
| \(\phi_{ld}, \phi_{ld}\) | 0.19 | 0.18 |
| \(\phi_{ow}, \phi_{ld}\) | 0.20 | 0.15 |
| \(\phi_{ow}, \phi_{ow}\) | 0.21 | 0.19 |

### Table VI

| Comparison Pair | CoRes | BCT |
|-----------------|-------|-----|
| \(\phi_{ld}, \phi_{ld}\) | 0.59 | 0.63 |
| \(\phi_{ow}, \phi_{ld}\) | 0.65 | 0.54 |
| \(\phi_{ow}, \phi_{ow}\) | 0.69 | -  |
Fig. 10. Feature spatial configuration with 2-step upgrading (1 class per upgrade) and MNIST dataset with 2D features. Colored cloud points are the features from the test-set and lines represent the classifier prototypes. Compatibility methods compared: (a) IFT; (b) BCT; (c) CoReS.

VI. ABLATION STUDIES

As CoReS is a single building block method, ablation study consists in tuning training hyperparameters. In the following, we present experiments to evaluate the factors that affect the performance of our training procedure. In particular: in Section VI-A, we discuss how much the number of training epochs affects learning compatible representations; in Section VI-B, we evaluate the influence of Model Selection over compatibility; in Section VI-C, we analyze the effects of starting from a pre-trained model; in Section VI-D, we examine model initialization, whether using same and different random initialization or fine-tuning from a previously learned model; in Section VI-E, we consider the effect of using different class sequence ordering; in Section VI-F we study how much different network architectures impact compatibility; in Section VI-G, we examine whether the number of pre-allocated future classes in the $d$-Simplex classifier impacts the performance of CoReS. Finally, in Section VI-H, we explore whether different source of upgrade data influences the compatibility performance.

The ablation experiments are performed on the verification task of Section V-A using CIFAR-100/10 dataset with nine upgrades.

A. Number of Epochs

In this experiment, models were learned sequentially with 30, 70, 100, 200 and 350 epochs at each upgrade with 9-step upgrading. We modified the learning rate as follows: no change for the 30-epochs case; scheduled to decrease to 0.1 at epoch 50 for the 70-epochs case; scheduled to decrease to 0.01 at epoch 70 for the 100-epochs case (already discussed in Section V-A); scheduled to decrease to 0.1 at epoch 70 and 0.01 at epoch 140 for the 200 epochs case; scheduled to decrease to 0.1 at epoch 150 and to 0.01 at epoch 250 for the 350 epochs case.

Fig. 11 shows plots of AC (Fig. 11(a)) and AM (Fig. 11(b)) for this experiment. We can notice that the trade-off between AC and AM occurs near 100 epochs. As AC decreases, the AM loses its meaning as a metric for compatibility evaluation.

B. Model Selection

In Section IV-C, we considered the case in which at each upgrade the training-set is increased in size and extended the basic CoReS formulation introducing compatibility learning with Model Selection. Table VII compares AC and AM of the basic CoReS implementation against CoReS with Model Selection on CIFAR-100 and nine upgrade steps in 20 runs. It can be observed that Model Selection provides some improvement of AC while keeping the same AM. A larger improvement is observed when Model Selection combined with pre-training as discussed in the next subsection.

C. Pre-Training

In this section we analyze the effect of pre-training over compatibility. We reserved 50 classes of CIFAR-100 to pre-train the model and 50 classes to create 10 distinct training-sets to upgrade the model. The 50-classes pre-trained model was used for initialization. Five classes were added at each upgrade. Since the size of pre-training changes over the upgrades, we...
expected to observe the correlation between pre-training and compatibility.

The values of $AM$ and $AC$ in relationship with the number of classes leveraged for pre-training are reported in Table VIII for CoReS without and with Model Selection over 20 runs. From the table, it appears that the more data are used for pre-training the more the model improves the $AC$ and $AM$. The Model Selection strategy provides an additional increase of $AC$ of about 8% at any level of pre-training with nearly the same improvement of $AM$.

For the sake of comparison, we implemented pre-training with Model Selection also on BCT in the same test scenario. The compatibility matrices of CoReS and BCT are shown in Fig. 12(a) and (b), respectively. We see that CoReS achieves compatibility in almost all the upgraded models, achieving an $AC$ of 0.78, with an evident large improvement over BCT.

### D. Model Initialization

In our previous experiments, the network was trained from scratch at each upgrade, using the same, randomly selected, initial parameters. According to this, all the upgraded models started optimization from the same configuration of parameters. Alternatively, parameters can be randomly selected at each upgrade. As a third option, learning at each upgrade can also be performed by starting from the previous upgrade and then performing fine-tuning incrementally.

Table IX reports mean and standard deviation of $AC$ and $AM$ for 9-step upgrading and 20 runs for these three cases (SAME, RANDOM, and FINE-TUNED, respectively). It can be noticed that random initialization at each upgrade negatively affects the final performance, resulting in a sensible reduction of $AC$ with respect to initialization with the same parameters at each upgrade. This result can be supported by the observations in [69], [70] and is likely related to the concept of “flat minima” [71], [72] and to the fact that with the same initialization, the SGD optimization explores the same basin of the loss landscape near the minimum. We argue that the basins of the loss landscape near the minimum explored by CoReS across upgrades has a close relationship with the stationarity/compatibility of the learned representation. On its turn, FINE-TUNED results in a strong reduction in $AC$. We argue that this is due to the fact that in this case, differently from learning from scratch, optimization should go back to earlier weight configurations to find more complex error landscapes.
TABLE X
COMPATIBILITY OF CoRES AS A FUNCTION OF SEQUENTIAL CLASS ORDERING WITH 9-STEP UPGRADING OVER 20 RUNS: WITH ALPHABETICAL ORDER (ALPHABETICAL); WITH DIFFERENT RANDOM PERMUATION OF THE CLASSES AT EACH RUN (RANDOM). MEAN AND STANDARD DEVIATION OF $AC$ AND $AM$

| CLASS ORDERING | $AC$     | $AM$     |
|----------------|----------|----------|
| ALPHABETICAL   | 0.51 ± 0.10 | 0.59 ± 0.08 |
| RANDOM         | 0.52 ± 0.13 | 0.57 ± 0.05 |

E. Different Class Order

Table X reports mean and standard deviation of $AC$ and $AM$ of 20 runs for different sequential class orderings, namely: (1) classes are alphabetically ordered, (ALPHABETICAL); (2) each run has a different random permutation of the classes (RANDOM), which we adopted as default choice in our experiments. It can be noticed, $AC$ and $AM$ metrics are almost the same in the two cases showing that class ordering has essentially no effect on learning compatible features.

F. Different Model Architectures

It has been shown that the expressive power of the network architecture has impact on the classification accuracy of fixed classifiers since the complexity of learning compatible representations is fully demanded to the internal layers of the neural network [27], [53].

According to this, it is relevant to evaluate the impact of the expressive power of network architectures over CoReS compatible learning. We compared different architectures with increasing expressive power, namely: ResNet20 (0.27 M parameters), ResNet32 (0.46 M parameters), SENet-18 (1.23 M parameters) and RegNetX_200MF (2.36 M parameters). We set the SGD initial learning rate to 0.1 and the weight decay factor to $5 \times 10^{-4}$. ResNet20 was trained for 40 epochs with constant learning rate. ResNet32 was trained for 70 epochs, with learning rate reduced to 0.01 at epoch 50, and to 0.001 at epoch 65. RegNetX_200MF was trained for 150 epochs, with learning rate reduced to 0.01 at epoch 70 and to 0.001 at epoch 120.

Fig. 13(a), (b), (c) and (d) show the CoReS compatibility matrices with ResNet20, ResNet32, SENet-18, and RegNetX_200MF respectively. It can be noticed that both compatibility and verification accuracy follow the expressive power of the network model. ResNet32 in Fig. 13(b) has compatibility similar to Fig. 13(a), but the verification accuracy is higher. RegNetX_200MF improves both compatibility and verification accuracy Fig. 13(d). This experiment also confirms the general applicability of our compatibility learning approach.

Finally, we analyzed the practical case in which a deployed system is upgraded not only with fresh data but also with recent and more powerful network architectures. To this end, we evaluate the case in which the ResNet20 is first upgraded with SENet-18, and subsequently with RegNetX_200MF. Fig. 14 reports the compatibility matrix for this experiment with the CIFAR-100/10 datasets. As can be observed, the learned representations remain compatible as the network architecture is upgraded. The increasing expressive power of the architectures also positively reflects on compatibility.

G. Different Number of Future Classes

It is relevant to study how the number of future classes of (9) influences the compatibility of a learned representation model. In Table XI, we evaluate CoReS with a different number of future classes with one, two, four and nine-step upgrades. As can be noticed from the table, performance does not vary as the number of future classes increases. The $AC$ stabilizes as the number of upgrades increases. The $AC$ may have sudden variations when the number of upgrades is small as it basically averages the number of times compatibility is achieved.

A related ablation for the classification task in Class-incremental Learning has already been studied in [25]. CiL addresses incremental classification under catastrophic forgetting. Given the substantial difference between the classification and

7.Code source for ResNet20 and RegNetX_200MF can be found at https://github.com/kuangliu/pytorch-cifar and for ResNet32 at https://github.com/arthurducillard/incremental_learning.pytorch
TABLE XI
Ablation Study on CIFAR100 for Different Number of Future Classes With 1, 2, 4, 9-Step-Upgrading

| #FUTURE CLASSES | One-upgrade | Two-upgrades | Four-upgrades | Nine-upgrades |
|-----------------|-------------|--------------|---------------|---------------|
|                 | AC AM AC AM | AC AM AC AM | AC AM AC AM | AC AM AC AM |
| 100             | 1           | 0.62 0.67 0.62 0.9 | 0.59 0.51 0.59 |               |
| 200             | 1           | 0.68 1 0.66 0.8 | 0.62 0.53 0.57 |               |
| 500             | 1           | 0.67 1 0.64 0.7 | 0.64 0.51 0.60 |               |
| 1000            | 1           | 0.66 0.67 0.63 0.9 | 0.62 0.56 0.55 |               |
| 10000           | 1           | 0.63 1 0.62 0.7 | 0.59 0.51 0.56 |               |

TABLE XII
Compatibility of CoReS for Training With Different Source of Upgrade Data With 1-Step-Upgrading: From New Classes (new classes); From Old Classes (old classes); From Both New and Old Classes (old & new classes)

| COMPARISON PAIR | NEW CLASSES | OLD CLASSES | OLD & NEW CLASSES |
|-----------------|-------------|-------------|-------------------|
|                 | M ECC Γ (%) | M ECC Γ (%) | M ECC Γ (%) |
| #(φold, φ1d)    | 0.60 - - 0.64 - - 0.59 - - |
| #(φnew, φ1d)    | 0.61 √ 21.3 0.65 √ 33.3 0.64 √ 59.6 |
| #(φnew, φnew)   | 0.65 - - 0.66 - - 0.66 - - |

VII. Conclusion
In this paper, we presented CoReS, a new training procedure that provides compatible representations grounding on the stationarity of the representation as provided by fixed classifiers. In particular, we exploited a special class of fixed classifiers where the classifier weights are fixed to values taken from the coordinate vertices of regular polytopes. They define classes that are maximally separated in the representation space and maintain their spatial configuration stationary as new classes are added. With our solution, there is no need to learn any mappings between representations nor to impose pairwise training with the previously learned model. The functional complexity is fully demanded to the internal layers of the network. We analyzed and discussed the factors that affect the performance of our method and performed extensive comparative experiments on visual search applications of different complexity. We demonstrated that our solution for compatibility learning based on feature stationarity largely outperforms the current state-of-the-art and is particularly effective in the case of multiple upgrades of the training-set, that is the typical case in real applications. CoReS also provides large compatibility when the network architecture is upgraded to newer and more powerful models.

REFERENCES
[1] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., 2005, pp. 539–546.
[2] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
[3] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “CNN features off-the-shelf: An astounding baseline for recognition,” in Proc. IEEE Comput. Vis. Pattern Recognit. Workshops, 2014, pp. 806–813.
[4] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” in Proc. Adv. Neural Inf. Process. Syst., Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds, Curran Associates, Inc., 2014, pp. 3320–3328.
[5] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” in Proc. IEEE Comput. Vis. Pattern Recognit., 2015, pp. 815–823.
[6] W. Liu, Y. Wen, Z. Yu, and M. Yang, “Large-margin softmax loss for convolutional neural networks,” in Proc. 33rd Int. Conf. Mach. Learn., 2016, pp. 507–516.
[7] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “DeepFace: Closing the gap to human-level performance in face verification,” in Proc. IEEE Comput. Vis. Pattern Recognit., 2014, pp. 1701–1708.
[8] Y. Sun, Y. Chen, X. Wang, and X. Tang, “Deep learning face representation by joint identification-verification,” in Proc. Adv. Neural Inf. Process. Syst., Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2014, pp. 1988–1996.
[9] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “ArcFace: Additive angular margin loss for deep face recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 4690–4699.
[10] D. Yi, Z. Lei, S. Liao, and S. Z. Li, “Deep metric learning for person re-identification,” in Proc. 22nd Int. Conf. Pattern Recognit., 2014, pp. 34–39.
[11] W. Li, R. Zhao, T. Xiao, and X. Wang, “DeepReID: Deep filter pairing neural network for person re-identification,” in Proc. IEEE Comput. Vis. Pattern Recognit., 2014, pp. 152–159.
[12] L. Zheng, H. Zhang, S. Sun, M. Chandraker, Y. Yang, and Q. Tian, “Person re-identification in the wild,” in Proc. IEEE Comput. Vis. Pattern Recognit., 2017, pp. 1367–1376.
[13] T. Chen et al., “ABD-Net: Attentive but diverse person re-identification,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 8351–8361.
[14] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, “Neural codes for image retrieval,” in Proc. Eur. Conf. Comput. Vis., Springer, 2014, pp. 584–599.
[65] J. Johnson, M. Douze, and H. Jégou, “Billion-scale similarity search with GPUs,” IEEE Trans. Big Data, vol. 7, no. 3, pp. 535–547, Jul. 2021.

[66] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, “Deep gradient compression: Reducing the communication bandwidth for distributed training,” in Proc. Int. Conf. Learn. Representations, 2018.

[67] L. Jing, P. Vincent, Y. LeCun, and Y. Tian, “Understanding dimensional collapse in contrastive self-supervised learning,” in Proc. Int. Conf. Learn. Representations, 2021.

[68] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in Proc. Int. Conf. Mach. Learn., PMLR, 2020, pp. 1597–1607.

[69] V. Nagarajan and J. Z. Kolter, “Uniform convergence may be unable to explain generalization in deep learning,” in Proc. Adv. Neural Inf. Process. Syst., H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, Art. no. 1042.

[70] B. Neyshabur, H. Sedghi, and C. Zhang, “What is being transferred in transfer learning?,” in Proc. 34th Int. Conf. Neural Inf. Process. Syst., 2020, Art. no. 44.

[71] S. Hochreiter and J. Schmidhuber, “Simplifying neural nets by discovering flat minima,” in Proc. Adv. Neural Inf. Process. Syst., 1995, pp. 529–536.

[72] S. Hochreiter and J. Schmidhuber, “Flat minima,” Neural Comput., vol. 9, no. 1, pp. 1–42, 1997.

Niccolò Biondi received the MS degree (cum laude) in computer engineering from the University of Firenze, Italy, in 2021. Presently, he is working toward the PhD degree with the University of Firenze at MiCC, Media Integration and Communication Center, University of Firenze. His research interests include machine learning and computer vision with special focus on compatible learning, representation learning, and incremental learning.

Federico Pernici received the laurea degree in information engineering, the post-laurea degree in Internet engineering, and the PhD degree in information and telecommunication engineering from the University of Firenze, Italy, in 2002, 2003, and 2005, respectively. He is assistant professor with the University of Firenze, Italy. He has been working with the MiCC Media Integration and Communication Center as a research assistant since 2002, and has also served as adjunct professor with the University of Firenze. His scientific interests are computer vision and machine learning with a focus on different aspects of visual tracking, incremental learning and representation learning. He was associate editor of Machine Vision and Applications Journal.

Matteo Bruni received the MS degree (cum laude) in computer engineering from the University of Firenze, Italy, in 2016, and the PhD degree in information engineering from the University of Firenze, in 2020. His research interests include pattern recognition and computer vision with specific focus on feature embedding, face recognition, incremental learning, and compatible representation.

Alberto Del Bimbo (Senior Member, IEEE) is full professor with the University of Firenze, Italy. He is the author of more than 350 scientific publications in computer vision and multimedia and Principal investigator of technology transfer projects with industry and governments. He was the program chair of ICPR 2012, ICPR 2016 and ACM Multimedia 2008, and the general chair of IEEE ICICCS 1999, ACM Multimedia 2010, ICPR 2011, ECCV 2012 and ICPR2020. He is the general chair of ACM Multimedia 2021. He is the editor in chief of ACM TOMM Transactions on Multimedia Computing Communications and Applications and associate editor of Multimedia Tools and Applications and Pattern Analysis and Applications journals. He was associate editor of IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Multimedia and Pattern Recognition and also served as the guest editor of many Special Issues in highly ranked journals. He is the chair of ACM SIGMM the Special Interest Group on Multimedia. He is IAPR fellow and ACM distinguished scientist and is the recipient of the 2016 ACM SIGMM Award for Outstanding Technical Contributions to Multimedia Computing Communications and Applications.