RSAC: Regularized Subspace Approximation Classifier for Lightweight Continuous Learning

Chih-Hsing Ho
Department of Electrical Engineering and Computer Science Honor Program
National Chiao Tung University
Hsinchu, Taiwan
bill86416.eecs04@nctu.edu.tw

Shang-Ho(Lawrence) Tsai
Department of Electrical Engineering
National Chiao Tung University
Hsinchu, Taiwan
shanghot@alumni.usc.edu

***Abstract*—Continuous learning seeks to perform the learning on the data that arrives from time to time. While prior works have demonstrated several possible solutions, these approaches require excessive training time as well as memory usage. This is impractical for applications where time and storage are constrained, such as edge computing. In this work, a novel training algorithm, regularized subspace approximation classifier (RSAC), is proposed to achieve lightweight continuous learning. RSAC contains a feature reduction module and classifier module with regularization. Extensive experiments show that RSAC is more efficient than prior continuous learning works and outperforms these works on various experimental settings.**

***Index Terms*—Continuous learning, Incremental Batch Learning, Streaming Learning**

I. Introduction

Deep networks have enabled significant advances over the last decade in many machine learning tasks, such as image classification [1]–[3] and object recognition [4]–[9]. The success is especially significant under the setting of supervised learning, where the entire labeled dataset is provided to train the deep network to complete the assigned task (i.e. image classification). Despite the success of supervised learning, its success is often achieved on the presumption that the tasks are assigned all at once. This is not a realistic learning procedure as human. As a realistic learner, human possess the ability to continually grow the knowledge throughout the lifespan by solving different tasks. The supervisions of those tasks from different time span assist the establishment of human capability [10], [11]. While human benefits from the continuous supervision and the shift of tasks from the environment, this is not the case for deep network, which fails on solving the old tasks when a new task is learned [12]–[14]. Such phenomenon has been referred as *catastrophic forgetting* [15]–[17] and its potential solutions are discussed in the literature of continuous learning [12], [14], [18], [19].

Continuous learning seeks to robustify the knowledge previously learned by deep network. By consolidating the knowledge, the network can be adopted in the scenario where data continuously streams in and achieves good performance on new coming tasks without forgetting the old ones. The mainstream solutions of performing continuous learning are knowledge consolidation [20]–[23], network expansion [24]–[27] and memory rehearsal [13], [28], [29]. While prior works provide several possibilities to avoid catastrophic forgetting, the training time is usually exhaustive [14], [30] and thus hinders the application of continuous learning in real world scenario, where the model not only has to perform well in the new tasks, but also has to complete the learning within a short period to keep track of the new coming data. Moreover, for the applications on edge computing, the memory usage is also another concern. In this paper, we consider these constraints and refer the problem as lightweight continuous learning (LCL).

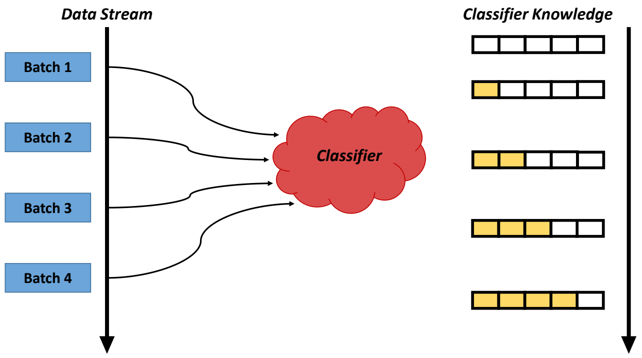

Fig. 1. A lightweight continuous learning algorithm is designed to perform the learning as data streams in under the constraint that both training and inference time is short and memory usage is low.

To address this problem, a novel algorithm, **regularized subspace approximation classification (RSAC)**, is proposed which contains a transformation module and a classifier module. For the transformation module, a lightweight feature reduction algorithm is introduced. This is inspired by prior works [31], [32] that discard the use of backpropagation in deep network and substitute with a set of explainable modules. For the classifier, the quadratic discriminant classifier (QDC) [33] is used with the proposed regularization. By combining these 2 modules, RSAC expedites the training under low memory usage without sacrificing the performance.

In summary, the contribution of this work is 3 folds. First, the current limitations of continuous learning methods, including large memory usage and long training time, are discussed. The problem of lightweight continuous learning (LCL) is then formulated and investigated. Second, we proposed a novel algorithm, regularized subspace approximation classifier (RSAC), for LCL problem. Finally, extensive experiments demonstrate that RSAC achieves state-of-the-art performance under different continuous learning settings with large computation time improvement. We hope our model can contribute to the real world continuous learning applications under various practical constraints.

II. Related work

In this section, the limitations of previous continuous learning works are discussed and the prior lightweight transformation and classification algorithms, which this work is inspired of, are reviewed.

A. Continuous learning

The study of continuous learning seeks to mitigate the catastrophic forgetting phenomenon [14], [21], [34], where the classifier forgets the knowledge previously established after training on new data. The proposed approaches can be mainly categorized into weight consolidation [20], [35], architecture expansion [24]–[27] and memory rehearsal [24], [25]. Their infeasibility on applying to lightweight continuous learning (LCL) scenario is discussed.

Weight consolidation based methods [35] impose constraint to avoid dramatic shift on the learned weights. The constraint encourages similar output activation between the use of the current weights and the updated weights. This can be implemented through knowledge distillation loss [20], L2 regularization [36] or updating the weight with smaller learning rate [37]. In addition, these constraints can be explicitly applied to those weights that are important to specific task [21]–[23]. However, these methods often suffer from insufficient learning capacity as the flexibility is restricted by the regularization [14] imposed for consolidating the old knowledge. Moreover, methods like LwF [20] are not applicable for LCL scenario as the training time increases proportionally with the number of tasks.

Expanding the architecture dynamically is another type of solutions for continuous learning and breaks the inflexibility limitation of weight consolidation based methods. The expansion can be performed by allocating new subnetwork [24]–[26], [38] or inserting neurons in a hierarchical manner [26]. Aside from adding completely new modules, useful neurons from trained feature extractor can be selected by leveraging dynamic path [27] or gating functions [39]. In addition, to prevent overfitting of the incrementally larger network, [40] merges neurons with similar response for downstream training. However, these approaches are difficult to scale up in general when new coming tasks increase dramatically and thus are not applicable to the LCL scenario.

Rehearsal based approaches [13], [41]–[44] have demonstrated recent success on continuous learning by training on few examples from the previous batches, which can be sampled from the storage with limit memory [13], [43] or from the deep generators [28], [44]–[46] (i.e. autoencoder [47] and generative adversarial network [48]). However, explicitly storing past examples requires extra memory usage and the use of deep generators needs additional training. Again, all these prior works do not meet the requirement of LCL, which emphasizes on short training time, memory efficient and fast inference.

B. Lightweight transformation and classification

Despite the recent success of deep networks, it requires large computation resources and training time, which hinders its real world applications especially on edge computing. These drawbacks are addressed with subspace approximation with augmented kernels (Saak) [32] and its variant, subspace approximation with adjusted bias (Saab) [31]. Saak transform is an interpretable one-pass feedforward network based on truncated Karhunen-Loève Transform (KLT) [49], or PCA [50], that transforms the input domain (i.e. spatial domain for images) into a latent domain to obtain its associated latent representation. In order to approximate functions with higher complexity that maps between the input and the latent representation, cascaded Saak transformations are used together with the ReLu function in between individual Saak transformation. However, the use of ReLu function sacrifices partial information as the value below zero are truncated. To minimize the loss of information, a negative counterpart of the kernel vectors from truncated KLT is introduced, such that all the information will be preserved after projecting on the kernel vectors from truncated KLT and its negative counterpart.

Despite the advantages of Saak transform, the size of latent representation will grow exponentially with more cascade of Saak, due to the use of additional kernel vectors. To overcome this drawback, subspace approximation with adjusted bias (Saab) [31] transformation adds a computed bias term on each of the projection on KLT kernel vectors. Thus, Saab not only inherits the advantages of Saak transform, but further improves the memory efficiency. In addition, Saak transform and Saab transform have demonstrated competitive results compared to deep neural network on different domains, including image classification [31], [32], [51], [52], 3D object classification [53] and texture analysis [54]. Moreover, they can be applied in semi-supervised learning [55] and image compression [56] and demonstrate robustness toward adversarial attacks [57]. Inspired by the characteristic of these transformations in terms of memory efficient, lightweight computation and success in multiple domains, we explore whether the same benefits ensue in the scenario of lightweight continuous learning.

III. Method

In this section, the proposed algorithm regularized subspace approximation classifier (RSAC) for lightweight continuous learning (LCL) is introduced.

A. Preliminary formulation

Given a labeled dataset $\mathcal{D}_{full} = \{x_i, y_i\}_{i=1}^{N}$, where $x_i \in \mathcal{R}^d$ is an image, $\mathcal{X}_{full} = \{x_i\}_{i=1}^{N}$ is a full image set, $y_i \in \mathcal{Y}_{full} =$

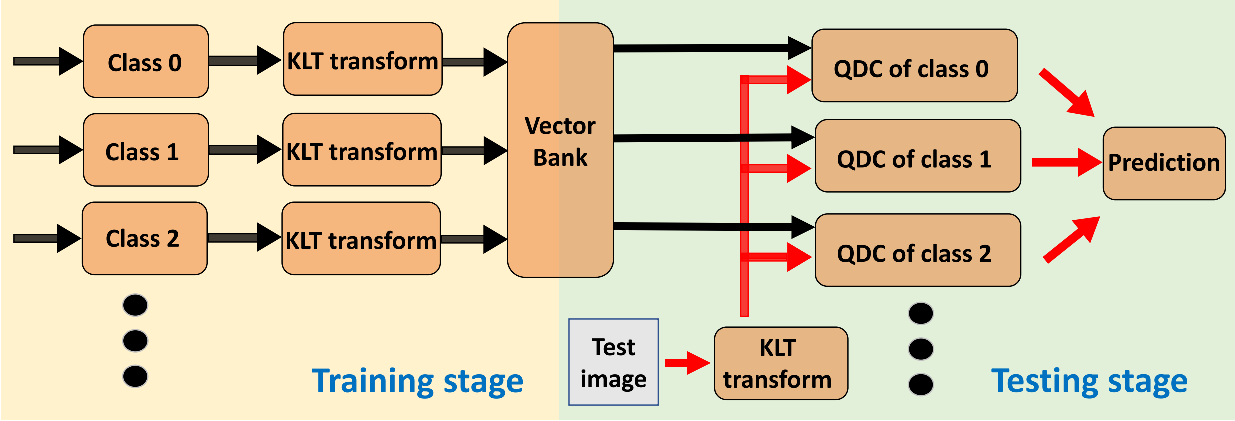

Fig. 2. The flow chart of the proposed RSAC architecture for lightweight continuous learning. During training stage, the eigenvectors of the KLT transform will be stored in the vector bank after learning. During inference stage, the dimension of data from each class will be reduced through KLT transform. The reduced feature will then be classified with the QDC classifier.

$\{1 \dots C\}$ is a full label set and C is the number of classes. A classifier F is trained to solve the assigned task originated from the dataset $\mathcal{D}_{full}$. Let $\mathcal{T}$ be the set of tasks to be solved by the classifier F. For a classification problem on image set $\mathcal{X} = \{x_i\}$ and label set $\mathcal{Y} = \{y_i\}$, the task $t(\mathcal{X}, \mathcal{Y})$ is defined to classify an image x in $\mathcal{X} \subseteq \mathcal{X}_{full}$ to the target label set $\mathcal{Y} \subseteq \mathcal{Y}_{full}$. In other words, the classifier aims to maximize the posterior class probability

$$P_{Y|X}(c|x) = \frac{P_{X|Y}(x|c)P_Y(c)}{P(X)}, \tag{1}$$

which can be expressed with the likelihood $P_{X|Y}(x|c)$ and class prior $P_Y(c)$ with Bayes rule [58].

For the supervised learning scenario, the entire labeled dataset are used during the training procedure, where $\mathcal{T} = \{t_1(\mathcal{X}_{full}, \mathcal{Y}_{full})\}$ as there is only a single task. While for continuous learning with M different tasks, the task set is defined as

$$\mathcal{T} = \{t_j(\mathcal{X}_j, \mathcal{Y}_j)\}_{j=1}^{M}, \tag{2}$$

and satisfies the constraint that

$$\mathcal{X}_i \cap \mathcal{X}_j = \emptyset, \quad i \neq j, \quad \bigcup_{j=1}^{M} \mathcal{X}_j = \mathcal{X}_{full}. \tag{3}$$

Correspondingly, the label set has to satisfy $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ and $\bigcup_{j=1}^{M} \mathcal{Y}_j = \mathcal{Y}_{full}$. Note that the general definition of task set $\mathcal{T}$ can be applied to supervised learning by setting $M = 1$.

B. Deep continuous learning

While the classifier F can be implemented in many different ways, one of the common manners is to leveraged the deep neural network model. Under the general definition of task set $\mathcal{T}$, the classifier can be formulated as $F : \mathcal{R}^d \rightarrow \mathcal{R}^C$. This is implemented by the combination of a feature extractor $f_\theta(x)$ of parameters θ and a softmax regression layer that predicts the posterior class probability as

$$F_c = P_{Y|X}(c|x) = \frac{e^{w_c^T f_\theta(x)}}{\sum_{k=1}^{C} e^{w_k^T f_\theta(x)}}, \tag{4}$$

where F_c denotes the c^{th} entry of classifier F, w_c is the vector of classification parameters of class c. However, the learning of θ is often not transparent and fails to yield an explainable result [59]. Moreover, the learning of θ requires the backpropagation procedure, which is computation expensive and relies on great amount of computing resources. All these drawbacks impede the advance of lightweight continuous learning (LCL) in real world applications.

C. Lightweight continuous learning

To avoid high computation cost, regularized subspace approximation classifier (RSAC) is proposed in this section as a solution for lightweight continuous learning (LCL) and contains 2 modules, including a feature reduction module and a classifier module.

1) Feature reduction module: Feature reduction is a critical stage in entire LCL pipeline, as simply taking raw image $x \in \mathcal{R}^d$ as input to the classifier will lead to high computational cost. For deep continuous learning, the feature reduction is performed by the feature extractor f of (4), while the KLT transformation is leveraged in lightweight continuous learning, inspired by the lightweight Saab transformation used in supervised learning. KLT is established on the covariance matrix Σ_c of class c computed by

$$\mu_c = \frac{1}{N_c}\sum_{j=1}^{N_c} x_j, \tag{5}$$

and

$$\begin{aligned} \Sigma_c &= \frac{1}{N_c}\sum_{j=1}^{N_c}(x_j - \mu_c)(x_j - \mu_c)^T \\ &= Q_c \Lambda_c Q_c^T, \end{aligned} \tag{6}$$

where N_c is the number of data belongs to class c, Λ_c is a diagonal matrix with the eigenvalue σ_c^j as the j^{th} entry and Q_c is a $d \times d$ orthonormal matrix, where the j^{th} column vector is the eigenvector q_c^j. The feature reduction is then implemented by projecting on the top k eigenvectors in Q_c correspond to the k largest eigenvalues in Λ_c, which are selected as

$$\frac{\sum_{j=1}^{k} \sigma_c^j}{\sum_{j=1}^{d} \sigma_c^j} \geq t, \tag{7}$$

where t is a power threshold to guarantee sufficient information is preserved. Let $\tilde{Q}_c$ to be a $d \times k$ matrix, where the column vectors are the concatenation of k selected eigenvectors. The resulting latent representation after projection is denoted as $f(x) = \tilde{Q}_c^T x$. Note that unlike the feature extractor f in (4), no parameter is needed to be learned in feature reduction module. Moreover, the power threshold t can be used to control the number of eigenvectors stored in the vector bank with limited memory buffer, as shown in the training stage of Fig2. With such projection scheme, the input image x can be represented with $f(x) \in \mathcal{R}^k$.

2) Classifier module: To discard the learning of numerous parameters in deep neural network, the relationship of (4) and (1) is investigated. We then note that maximizing the class posterior probabilities of (1) can be reformulated as

$$\max P_{Y|X}(c|x) = \max P_{X|Y}(x|c) P_Y(c), \tag{8}$$

by dropping the denominator $P(x)$ in (1), since the distribution of the input data X is independent of the learning. In general, the class conditional distribution $P_{X|Y}(x|c)$ can be modeled with any function in the exponential family distribution [60], i.e.

$$P_{X|Y}(x|y) = q(x) e^{<w_y, v(x)> - \phi(w_y)} \tag{9}$$

and

$$P_Y(y) = \frac{e^{\phi(w_y)}}{\sum_{i=1}^{C} e^{\phi(w_i)}}, \tag{10}$$

where w_y is a canonical parameter, $v(x)$ is a sufficient statistic, $\phi(w_y)$ is a cumulant function and $q(x)$ is a underlying measure [60]. While, in principle, any function in the exponential family can be leveraged, a simple Guassian distribution is considered in this work. The likelihood that operates on the latent representation $f(x) \in \mathcal{R}^k$ from feature module is then modeled with

$$\begin{aligned} P_{X|Y}(f(x)|c) &= \mathcal{G}(f(x); \hat{\mu}_c, \hat{\Sigma}_c) \\ &= \frac{1}{(2\pi)^{d/2} |\hat{\Sigma}_c|} e^{-\frac{1}{2}(f(x) - \hat{\mu}_c)^T \hat{\Sigma}_i^{-1} (f(x) - \hat{\mu}_c)}. \end{aligned} \tag{11}$$

Similarly, the class prior can be computed as

$$P_Y(c) = \frac{N_c}{\sum_{j=1}^{C} N_j}. \tag{12}$$

Note that the computation of $\hat{\mu}_c$ and $\hat{\Sigma}_c$ of class c is similar to (5) and (6), but operates on the latent representation $f(x)$.

Algorithm 1 Pseudocode of RSAC

```
EigList:= []
while Training do
    while Data $\mathcal{X}_c$ from new class $c$ stream in do
        Apply feature reduction module in Sec. III-C1.
        Append the $k$ eigenvectors of class $c$ selected from
        (7) to EigList.
    end while
end while
while Testing do
    for all test images $x$ do
        Apply feature reduction module in Sec. III-C1.
        Perform QDC of (14) with regularization of (16) on
        the latent feature.
    end for
end while
```

As shown in the inference stage of Fig 2, an input example x is mapped to the latent representation $f(x)$ with the feature reduction module and then classified by computing the maximum a posteriori as

$$\arg\max_c P_{Y|X}(c|f(x)) = \arg\max_c P_{X|Y}(f(x)|c) P_Y(c). \tag{13}$$

By substituting the modeling of (11) and (12) into (13), it can be re-written in log scale as

$$\begin{aligned} &\arg\max_c ln(P_{Y|X}(c|f(x))) = \arg\max_c ln(P_{X|Y}(f(x)|c)) + ln(P_Y(c)) \\ &= \arg\max_i ln(\frac{1}{|\hat{\Sigma}_c|}) - \frac{1}{2}(f(x) - \hat{\mu}_c)^T \hat{\Sigma}_c^{-1} (f(x) - \hat{\mu}_c) + ln(\frac{N_c}{\sum_{j=1}^{C} N_j}) \end{aligned} \tag{14}$$

This is referred as quadratic discriminant classifier (QDC) in the following.

By combining the feature and classifier module, the overall training scheme is referred as **subspace approximation classification (SAC)**. As summarized in Algorithm 1 and visualized in Fig.2, SAC takes a task from the LCL problem as input. Given a task t_j in (2), the feature reduction module maps the inputs associated to t_j into a k dimension latent representation, where the label dependent eigenvectors are learned and stored vector bank. During inference, the feature module is again applied to the input and the output latent representation is then passed through the QDC classifier for obtain final prediction.

3) Efficient learning with regularization: While SAC provides an efficient classification solution for the LCL problem, it can be problematic, because the inverse of $\hat{\Sigma}_c$ in (11) is an ill-defined problem numerically if it is close to a singular matrix. To solve the problem, the relationship between Σ_c and $\hat{\Sigma}_c$ is revisited. Since the latent representation $f(x) = \tilde{Q}_c^T x \in \mathcal{R}^k$, the covariance of the latent representation can be reformulated as

$$\begin{aligned} \hat{\Sigma}_c &= Cov(f(x)) = \tilde{Q}_c^T Cov(x) \tilde{Q}_c \\ &= \tilde{Q}_c^T \Sigma_c \tilde{Q}_c = \tilde{Q}_c^T Q_c \Lambda_c Q_c^T \tilde{Q}_c = \Lambda_c. \end{aligned} \tag{15}$$

Methods	Datasets (Accuracy) Mnist	KMnist	Fashion Mnist	Datasets (Training Time (sec)) Mnist	KMnist	Fashion Mnist
DGR [28]	90.44±1.56	69.25±2.94	74.83±5.50	315.99±2.25	748.75±51.17	760.21±21.72
DGR+distill [20], [28]	92.31±0.74	64.42±1.12	76.03±4.12	314.12±12.79	819.52±14.52	800.81±3.69
EWC [21]	20.45±1.15	19.54±0.12	19.97±0.02	398.86±11.04	719.89±21.95	697.24±53.39
Online EWC [61]	20.69±1.53	19.54±0.12	19.97±0.03	371.87±12.35	665.04±3.40	692.49±29.20
iCaRL [13]	93.24±0.70	70.83±2.78	79.61±0.79	200.16±9.83	468.38±4.98	466.60±11.09
LwF [20]	20.98±0.85	20.16±0.24	19.42±2.54	198.40±9.09	495.62±31.48	499.49±8.77
RtF [46]	93.75±1.28	66.16±3.06	74.11±4.82	253.37±9.22	639.66±25.56	678.42±34.04
SI [22]	19.85±0.10	19.53±0.09	19.97±0.02	194.16±87.6	503.72±5.15	498.37±3.28
CNDPM [62]	93.54±0.13	74.35±1.4	44.62±2.1	> 3600	> 3600	> 3600
Saak [32]	95.21	76.25	73.51	> 3000	> 3000	> 3000
Ours	**95.59**	**77.35**	**80.32**	**5.90**	**5.72**	**5.48**

TABLE I
COMPARISON WITH BASELINES UNDER CLASS INCREMENTAL LEARNING SCENARIO.

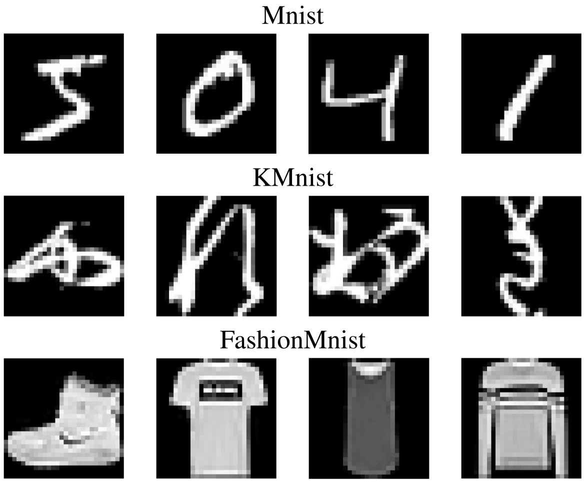

Fig. 3. Visualization of the examples in 3 datasets.

To avoid singularity, we proposed to add a regularization on $\hat{\Sigma}_i$ of (11) as

$$\hat{\Sigma}'_c = \hat{\Sigma}_c + \alpha * I = \Lambda_c + \alpha * I, \quad \forall c \in \mathcal{Y} \tag{16}$$

when singular matrix occurs and this is referred as **regularized SAC (RSAC)**. The use of RSAC not only avoids the singularity, but also simplifies the computation of (11) as the inversion of $\hat{\Sigma}'_c$ is also a diagonal matrix composed of the eigenvalues from Λ_c. Note that RSAC does not require large number of parameter learning and is more computation and memory efficiency for the lightweight continuous learning applications.

IV. Experiment

In this section, Mnist [63], KMnist [64] and FashionMnist [65] are evaluated in terms of averaged classification accuracy over all classes. Examples from all three datasets are visualized in Fig. 3. For all three datasets, there are 10 classes, 60000 training images and 10000 testing images. Unlike KMnist and FashionMnist, Mnist has a slightly imbalance data distribution across classes.

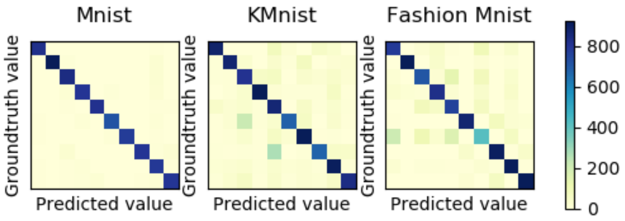

Fig. 4. Confusion matrix of Mnist, KMnist, FashionMnist from the QDC classifier.

Nine different continuous learning baselines are compared. The official code [1] of CNDPM [62] is adopted and the public available code[2] for the rest of the baselines is used. Note that while the original architecture of CNDPM is used, the rest of the baselines are implemented based on multilayer perceptrons (MLP) as backbones. In addition, although the Saak architecture is not designed specifically for continuous learning scenario, it can be compared with slight modification of its official code[3]. To avoid the memory explosion, we implement three stages Saak transform with transformation sizes 8 × 8, 2 × 2 and 2 × 2, and use KLT transform and regularized QDC for classification in order to fit the continuous learning scenario. For a fair comparison, the number of feature selected is the same between RSAC and Saak.

For continuous learning settings, 5 tasks are considered by separating the dataset into pairs (i.e. grouping class (0/1), (2,3) etc.). The regularization hyperparmeter α of (16) is set as 0.4 for all datasets.

A. Compare with continuous learning baselines

The continuous learning baselines are compared from the perspectives of classification accuracy and training time, as these 2 factors are critical to perform lightweight continuous learning (LCL). As shown in left of Table I, the proposed framework outperforms the continuous learning baselines both in terms of accuracy and training time. The accuracy gain

[1] https://github.com/soochan-lee/CN-DPM
[2] https://github.com/GMvandeVen/continual-learning
[3] https://github.com/davidsonic/Saak-Transform

Power threshold t	Mnist		KMnist		Fashion Mnist	
	k	acc	k	acc	k	acc
0.8	31	67.75	64	61.30	26	64.90
0.9	68	93.22	126	76.16	77	73.98
0.95	121	95.41	211	77.13	156	79.74
0.96	141	95.43	243	76.87	185	80.25
0.97	168	95.43	285	74.84	224	73.56
0.98	206	91.66	346	75.09	278	73.95
Best	150	95.59	192	77.35	183	80.32

TABLE II
DIFFERENT POWER THRESHOLDS t OF (7) ARE EXPLORED. EACH THRESHOLD CORRESPONDS TO A SPECIFIC k VALUE IN (7). THE k VALUE ASSOCIATED TO THE BEST ACCURACY OF THREE DATASETS ARE REPORTED.

is more than 0.38%, 1.10% and 0.71% on Mnist, KMnist and FashionMnist respectively. The corresponding confusion matrix of the proposed architecture is visualized in Fig. 4.

In addition to the accuracy gain, the training time of proposed framework is significantly smaller than those baselines implemented with deep network, as shown in right of Table I. Such efficiency is attributed to the discard of backpropagation during training and the prevention of memory explosion. Note that most of the continuous learning baselines, besides CNDPM, are implemented with 4 layers MLP in order to provide more competitive baselines, but the proposed framework still beats those baselines with significant margin without sacrificing the accuracy. The results presented in Table I demonstrate the efficiency and effectiveness of the proposed algorithm and suggest that the proposed framework is a more suitable solution for LCL problem.

B. Ablation study

In this section, the ablation study of the proposed framework is conducted by investigating the effect of power threshold t of (7) and the number of training data provided to the proposed framework.

1) Effect of power threshold: From Table II, it can be observed that the accuracy saturates when the power threshold t is around 0.95 for all three datasets. The benefit of adding more eigenvectors is marginal when $t > 0.95$. Moreover, when adding eigenvectors associated with small eigenvalues, the computation of QDC (14) will often lead to numerical error as the covariance matrix is not invertible. The proposed regularization of (16) can avoid such ill-defined inverse matrix with minor drop of accuracy.

Furthermore, while the input has 784 dimension, it is not necessary to store all 784 eigenvectors in the vector bank. Instead, the best accuracy reported in Table II shows that storing less than 200 eigenvectors per class is sufficient to achieve competitive performance. With such memory efficiency, it allows the edge device to classify more data from more classes under same amount of memory budget.

2) Effect of dataset size: It is known that the deep neural network requires a large number of labeled data for learning a good classifier [66]. The dependency of large number of labeled data is investigated on the proposed method, as shown in Fig. 5. Unlike the heavy dependency of neural network based methods, the proposed framework is data efficiency as the accuracy remains fairly stable when more than 500 (800) images per class are used, which only requires less than 15%, 10% of the available training data in Mnist (KMnist). For a more challenging dataset Fashion Mnist, merely 50% of the available training data is needed to achieve competitive results. This indicates that the proposed framework is more suitable in the application where training data is scarce.

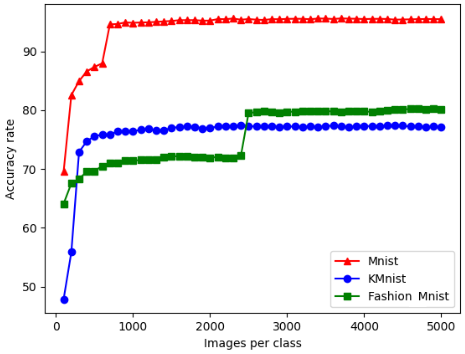

Fig. 5. Ablation study on the number of data needed for training a good LCL classifier. The accuracy saturates after the use of 15%, 10% and 50% images for Mnist, KMnist, Fashion Mnist, respectively.

Methods/Datasets	Mnist	KMnist	Fashion Mnist
Saak [32]	95.59	77.00	78.15
Ours	**95.92**	**80.48**	**80.24**

TABLE III
COMPARISON WITH SAAK UNDER DATA INCREMENTAL LEARNING SCENARIO.

C. Data incremental continuous learning

Inspired by the observation in the ablation study that the proposed method is less sensitive to the number of training data, the proposed method is further evaluated under a novel continuous learning scenario, where **data from same class** does not stream in the same time stamp. We refer this as *data incremental continuous learning*. Note that this is different from the typical continuous learning settings discussed in Sec. III-A, where data from same class always arrive at the same time stamp. Since most prior works in continuous learning literature does not fit into this scenario, the only comparable baseline is SaaK [32]. As shown in Table III, the proposed method beats SaaK for all 3 datasets. Moreover, the gain increases from 0.33% of simple dataset (i.e. Mnist) to 2.09% of a more challenging dataset (i.e Fashion Mnist).

V. CONCLUSION

In this paper, the underlying disadvantages of current continuous learning algorithms are discussed, including the slow training time, memory inefficiency and dependency on large number of training data. All factors harm the usability of continuous learning algorithms in the real world application, where the streaming data is limited and the training time is critical. As the result, the importance of lightweight continuous learning problem is investigated with the proposed algorithm regularized subspace approximation classifier (RSAC). RSAC inherits the advantages of previous lightweight transformation with new architecture to match continuous learning scenario. RSAC also consists a quadratic discriminant classifier (QDC) with addition regularization to prevent ill-defined condition. Moreover, the elaborate design of RSAC reduces the cost of memory storage and enables the classification to be perform in an efficient manner. Extensive experiments demonstrate the performance as well as training speed of the proposed framework. We hope this work will inspire the focus of lightweight continuous learning problem in the literature.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[3] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[4] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, 2013.

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[7] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928, 2015.

[8] Z. Wang, J. Xu, L. Liu, F. Zhu, and L. Shao. Ranet: Ranking attention network for fast video object segmentation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3977–3986, 2019.

[9] K. Duarte, Y. Rawat, and M. Shah. Capsulevos: Semi-supervised video object segmentation using capsule routing. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8479–8488, 2019.

[10] Jun Tani. *Exploring Robotic Minds: Actions, Symbols, and Consciousness as Self-Organizing Dynamic Phenomena*. Oxford University Press, Inc., USA, 1st edition, 2016.

[11] Andrew Bremner, David Lewkowicz, and Charles Spence. Multisensory development, 11 2013.

[12] Ian Goodfellow, Mehdi Mirza, Xia Da, Aaron Courville, and Y. Bengio. An empirical investigation of catastrophic forgeting in gradient-based neural networks. 12 2013.

[13] S. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5533–5542, 2017.

[14] German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *CoRR*, abs/1802.07569, 2018.

[15] A. Robins. Catastrophic forgetting in neural networks: the role of rehearsal mechanisms. In *Proceedings 1993 The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pages 65–68, 1993.

[16] Robert French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3:128–135, 05 1999.

[17] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks, 2017.

[18] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks, 2019.

[19] Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning, 2019.

[20] Zhizhong Li and Derek Hoiem. Learning without forgetting. *CoRR*, abs/1606.09282, 2016.

[21] James N Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114:3521 – 3526, 2017.

[22] Friedemann Zenke, Ben Poole, and Surya Ganguli. Improved multitask learning through synaptic intelligence. *CoRR*, abs/1703.04200, 2017.

[23] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. *CoRR*, abs/1711.09601, 2017.

[24] Jeongtae Lee, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *CoRR*, abs/1708.01547, 2017.

[25] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016.

[26] Tianjun Xiao, Jiaxing Zhang, Kuiyuan Yang, Yuxin Peng, and Zheng Zhang. Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *ACM Multimedia*, November 2014.

[27] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A. Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *CoRR*, abs/1701.08734, 2017.

[28] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *CoRR*, abs/1705.08690, 2017.

[29] Nicholas Ketz, Soheil Kolouri, and Praveen Pilly. Continual learning using world models for pseudo-rehearsal, 2019.

[30] Lorenzo Pellegrini, Gabriele Graffieti, Vincenzo Lomonaco, and Davide Maltoni. Latent replay for real-time continual learning, 2019.

[31] C.-C. Jay Kuo, Min Zhang, Siyang Li, Jiali Duan, and Yueru Chen. Interpretable convolutional neural networks via feedforward design. *Journal of Visual Communication and Image Representation*, 60, 03 2019.

[32] C. Kuo and Yueru Chen. On data-driven saak transform. *Journal of Visual Communication and Image Representation*, 50, 10 2017.

[33] Alaa Tharwat. Linear vs. quadratic discriminant classifier: an overview, 04 2016.

[34] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109 – 165. Academic Press, 1989.

[35] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[36] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *CoRR*, abs/1607.00122, 2016.

[37] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 647–655, Bejing, China, 22–24 Jun 2014. PMLR.

[38] Corinna Cortes, Xavi Gonzalvo, Vitaly Kuznetsov, Mehryar Mohri, and Scott Yang. Adanet: Adaptive structural learning of artificial neural networks. *CoRR*, abs/1607.01097, 2016.

[39] Nicolas Y. Masse, Gregory D. Grant, and David J. Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *CoRR*, abs/1802.01569, 2018.

[40] Guanyu Zhou, Kihyuk Sohn, and Honglak Lee. Online incremental feature learning with denoising autoencoders. In Neil D. Lawrence

and Mark Girolami, editors, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pages 1453–1461, La Palma, Canary Islands, 21–23 Apr 2012. PMLR.

[41] Tyler L. Hayes, Nathan D. Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. *CoRR*, abs/1809.05922, 2018.

[42] K. Lee, K. Lee, J. Shin, and H. Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 312–321, 2019.

[43] Francisco M. Castro, Manuel J. Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. *CoRR*, abs/1807.09536, 2018.

[44] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. *CoRR*, abs/1711.10563, 2017.

[45] Nitin Kamra, Umang Gupta, and Yan Liu. Deep generative dual memory network for continual learning. *CoRR*, abs/1710.10368, 2017.

[46] Michiel van der Ven and Andreas S. Tolias. Generative replay with feedback connections as a general strategy for continual learning. *CoRR*, abs/1809.10635, 2018.

[47] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *CoRR*, abs/1404.7828, 2014.

[48] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[49] Didier Lucor, Chau-Hsing Su, and George Em Karniadakis. Karhunen-loeve representation of periodic second-order autoregressive processes. In Marian Bubak, Geert Dick van Albada, Peter M. A. Sloot, and Jack Dongarra, editors, *Computational Science - ICCS 2004*, pages 827–834, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[50] Andrzej Maćkiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers Geosciences*, 19(3):303 – 342, 1993.

[51] Yueru Chen and C. C. Jay Kuo. Pixelhop: A successive subspace learning (ssl) method for object classification, 2019.

[52] Yueru Chen, Mozhdeh Rouhsedaghat, Suya You, Raghuveer Rao, and C. C. Jay Kuo. Pixelhop++: A small successive-subspace-learning-based (ssl-based) model for image classification, 2020.

[53] M. Zhang, H. You, P. Kadam, S. Liu, and C. . J. Kuo. Pointhop: An explainable machine learning method for point cloud classification. *IEEE Transactions on Multimedia*, pages 1–1, 2019.

[54] K. Zhang, H. Chen, Y. Wang, X. Ji, and C. . Jay Kuo. Texture analysis via hierarchical spatial-spectral correlation (hssc). In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 4419–4423, 2019.

[55] Y. Chen, Y. Yang, M. Zhang, and C. . J. Kuo. Semi-supervised learning via feedforward-designed convolutional neural networks. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 365–369, 2019.

[56] X. Zhang, S. Kwong, and C. . J. Kuo. Compressed image quality assessment based on saak features. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1730–1734, 2019.

[57] T. Ramanathan, A. Manimaran, S. You, and C. J. Kuo. Robustness of saak transform against adversarial attacks. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2531–2535, 2019.

[58] James V. Stone. *Bayes' Rule: A Tutorial Introduction to Bayesian Analysis*. Sebtel Press, 2013.

[59] Erico Tjoa and Cuntai Guan. A survey on explainable artificial intelligence (XAI): towards medical XAI. *CoRR*, abs/1907.07374, 2019.

[60] Rolf Sundberg. *Exponential Family Models*, pages 490–493. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[61] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress compress: A scalable framework for continual learning. In *ICML*, 2018.

[62] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. *ArXiv*, abs/2001.00689, 2020.

[63] L. Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[64] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature, 2018.

[65] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[66] Damien Brain and Geoffrey Webb. On the effect of data set size on bias and variance in classification learning. *Proceedings of the Fourth Australian Knowledge Acquisition Workshop*, 06 2000.