ARL-TR-9878 ● FEB 2024

Methodology of Soft Partition for Image Classification

by Vinod K Mishra and C-C Jay Kuo

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-9878 ● FEB 2024

Methodology of Soft Partition for Image Classification

Vinod K Mishra
DEVCOM Army Research Laboratory

C-C Jay Kuo
University of Southern California

REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		07/01/2023	12/31/2023
February 2024	Technical Report		

4. TITLE AND SUBTITLE
Methodology of Soft Partition for Image Classification

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
W911NF2020157		
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)
Vinod K Mishra and C-C Jay Kuo

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
DEVCOM Army Research Laboratory ATTN: FCDD-RLA-NC Aberdeen Proving Ground, MD 21005	ARL-TR-9878

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES
ORCID IDs: Vinod K Mishra, 0000-0001-9432-9082; C-C Jay Kuo, 0000-0001-9474-5035

14. ABSTRACT
The subspace learning machine (SLM) has been a powerful idea for machine learning and has been applied successfully to the task of image classification. Recently, a novel SLM method was proposed that (1) projects high-dimensional feature vectors into a 1D feature subspace and (2) partitions it into two disjoint sets. SLM with soft partitioning (SLM/SP) extends this approach by learning an adaptive soft decision tree structure using local greedy subspace partitioning. After meeting the stopping criteria for all child nodes and determining the tree structure, SLM/SP updates all projection vectors globally. SLM/SP enables efficient training, high classification accuracy, and a small model size. It is applied to experimental data to show its performance as a lightweight and high-performance classification method.

15. SUBJECT TERMS
Network, Cyber, and Computational Sciences; high-dimensional data; image classification; projection vectors; subspace learning

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	C. THIS PAGE		
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU	23

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
Vinod K Mishra	(410) 278-0114

STANDARD FORM 298 (REV. 5/2020)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

List of Tables

1. Introduction

The classic pattern recognition (PR) needs two modules, feature extraction and classification-oriented decision-making, operating in cascade. This report focuses on the latter module that can be done as either a single-stage (e.g., support-vector machine [SVM]) or multistage (e.g., decision tree [DT] and multilayer perceptron [MLP]) process. The subspace learning machine (SLM)[1,2] adopts DT architecture but, instead of a single feature, uses a new variable combining multiple features linearly at each node. The combination is the inner product of a projection vector (PV) and a feature vector. For a one-hot PV, the DT and SLM are identical.

Previous studies have reported on probabilistic and optimization-based searches as PV selection methods.[2] Both SLM and DT apply a hard split to a feature using a threshold at a decision node. Figure 1 shows the difference between hard and soft DT processes.

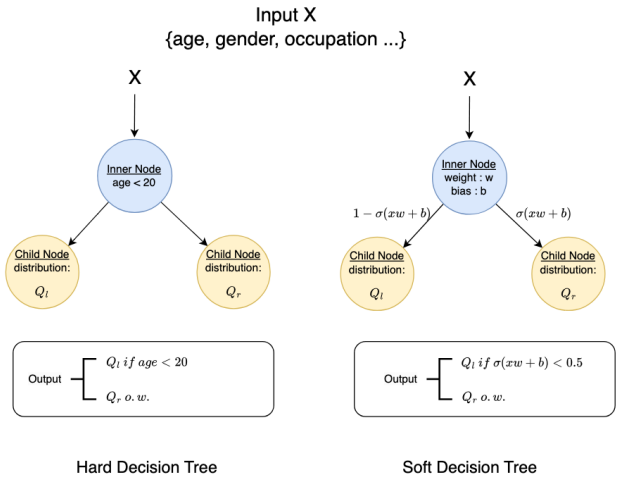

Fig. 1 The comparison of the decision process between a hard and a soft DT

We propose a new SLM method with soft partitioning (SP) using a soft DT data structure for decision-making. SLM/SP starts by learning an adaptive tree structure via local greedy subspace partitioning. Once the stopping criteria are met for all child nodes and the tree structure is finalized, all PVs are updated globally. This methodology enables efficient training, high classification accuracy, and a small model size. It is shown by experimental results that an SLM/SP tree offers a lightweight and high-performance classification solution. This report demonstrates the efficiency of SLM/SP in its model size and computational complexity.

2. Background

There are two types of learning methods for image classification:

1. Traditional machine learning methods, containing two cascaded modules: feature extraction from images and classification based on extracted features.

2. Deep learning (DL) methods, combining the two traditional modules into one. A convolutional neural network called AlexNet[3] pioneered this approach, and later models such as VGG, GoogLeNet, and ResNet made great improvements in accuracy. Further development of the vision transformer model[4] offers state-of-the-art (SOTA) performance for many image classification benchmarking datasets. Still, these methods suffer from problems like a lack of interpretability, high computational cost, and high model complexity.

2.1 Green Learning

Recently, the green learning (GL) approach to PR[5–9] has addressed several DL concerns (e.g., the high carbon footprints of large DL networks). GL has been shown to have low carbon footprint, small model size, low computational complexity, and logical transparency. It has also been successfully applied to point-cloud classification, segmentation, and registration[10,11]; fake image detection [12–15]; image generation[16–18]; blind image quality assessment[19]; disease classification[20]; face gender classification[21]; and object tracking.[22–24]

GL contains three cascaded modules: unsupervised representation learning, supervised feature learning, and supervised decision learning. The first two modules correspond to the feature extraction module in the classic PR paradigm. These modules are automated while feature extraction in PR is carried out by humans in an ad hoc manner. Module 3 usually deploys the XGBoost classifier, but we investigate the SLM/SP as its alternative.

In GL, automatic feature extraction is achieved by unsupervised representation learning method such as the Saab transform,[8] a joint spatial-spectral transform for decomposing a local patch into a direct-current and several alternating-current components based on the principal component analysis.

Multi-stage Saab transforms have been used to build several GL-based image classification methods (e.g., Pixelhop,[25] Pixelhop++,[26] and IPhop[27]). The Saab coefficients in various stages offer new representations for patches of different receptive fields and are used as feature candidates. They are obtained automatically

by exploiting the correlation of pixels in a local neighborhood without using supervision labels in their derivation. The dimension of the representation space is usually very large, so its dimension is reduced by a supervised learning method called the discriminant feature test (DFT).[28] To summarize, the multistage Saab transform serves as module 1 and DFT as module 2 of a GL system.

2.2 SLM

SLM outputs a PV of all possible image classes from a set of given discriminant features as input. It takes a tree structure, applies a subspace partitioning process for each node splitting, and generates purer leaf nodes for final predictions.

For subspace partitioning, SLM

1. identifies a discriminant subspace, denoted by S^0,
2. learns optimal PV from a set of candidates in S^0 to yield the most discriminant subspace, and
3. finds the optimal partitions for splitting the subspace of a parent node into those of child nodes.

The partitioning process is applied recursively at each child node to build an SLM tree. The training complexity of getting effective PVs is very high and it needs to be lowered substantially. Sec. III shows that the proposed SLM/SP method can significantly reduce it by replacing hard partitioning with soft partitioning.

2.3 Soft DT

The SLM/SP method uses ideas behind the soft DT[29,30] (Fig. 1). Some historical highlights of this approach are as follows:

1. The hierarchical mixture of experts[29] has parent nodes as linear classifiers with fixed-tree structure. Later soft DT work[30] examined a specific scenario, where axis-aligned features are used as the input and parent nodes are either static distributions over classes or linear functions.
2. A computationally efficient training method[31] was developed that optimized hard partitioning directly through differentiation with stochastic gradient estimators. More recent soft DTs[32,33] incorporate MLPs or convolutional layers in parent nodes to enable more complex partitioning of the input space.
3. Another approach combined nonlinear data transformations with DTs to enhance model performance (e.g., the neural decision forest[34] has achieved

SOTA performance on the ImageNet). A similar idea[35] used an MLP as the root transformer and to minimize a differentiable information gain loss.

However, it is important to point out that the model architectures are predetermined and fixed in all these methods. The choice of an effective architecture is still an open question.

3. SLM/SP Method

We start by considering a *K*-class classification problem, given an input feature space *X* containing *N* samples. Each sample has a *D*-dimensional feature vector denoted by

$$\boldsymbol{x}_n = (x_{n,1}, \dots, x_{n,d}, \dots x_{n,D})^T \in R^D, n = 1, \dots, N \quad (1)$$

The partitioning in the feature space in SLM can be expressed mathematically as

$$\boldsymbol{a}^T \boldsymbol{x} + b = 0 \quad (2)$$

where

$$\boldsymbol{a} = (a_1, \dots, a_d, \dots, a_D)^T, \left|\left|\boldsymbol{a}\right|\right| = 1 \quad (3)$$

is the unit vector (or PV) pointing in the surface normal direction and *b* is the bias. Then the full space, *S*, is split into two half subspaces:

$$S_+: a^T x \geq -b \text{ and } S_-: a^T x < -b \quad (4)$$

This process, used recursively, leads to a binary decision tree with a hierarchical partition of the feature space. This method is called the SLM since each node indicates a partitioned feature subspace. One challenge in Eq. 4 lies in finding a good PV $\boldsymbol{a}$ at each intermediate or inner node so that samples of different classes are better separated, as it is related to the sample distribution of different classes at the node. The ultimate objective is to lower the weighted entropy of all leaf nodes.

3.1 SLM/SP Approach

The SLM/SP combines soft partitioning with SLM. Let S_+ and S_- denote two child nodes as the output of the root node. In the usual approach, an input sample is assigned to one of the two child nodes S_+ and S_-, depending on the inequality. This is called hard partitioning. In SP, the probabilities of choosing either of the child nodes S_+ and S_- is given by

and

$$p_+(x) = \sigma(\boldsymbol{a}^T\boldsymbol{x} + b)$$
$$p_-(x) = 1 - \sigma(\boldsymbol{a}^T\boldsymbol{x} + b) \quad (5)$$

respectively, with σ as the sigmoid logistic function.

The complexity of SP depends on the dimension of $\boldsymbol{x}$, so for further improvement, the linear function $\boldsymbol{a}^T\boldsymbol{x} + b$ in Eq. 5 is replaced by a simple MLP. With nonlinear activation functions such as ReLU or Leaky ReLU, an MLP can be trained via back propagation. We use a single hidden-layer MLP as a preprocessing unit. Then, the probability of going to S_+ can be written as

$$p_+(x) = \sigma(\boldsymbol{a}'^T ReLU(\boldsymbol{a}^T\boldsymbol{x} + b) + \boldsymbol{b}') \quad (6)$$

The same SP process can be repeated at child nodes recursively. For example, we conduct soft partitioning on S_+ so the probability for an input to arrive at the $(+,+)$-grandchild node $p_{+,+}(x)$ is the cascaded multiplication of $p_+(x)$ at inner nodes. It is straightforward to get the mathematical expressions for $p_{+,+}(x)$, $p_{+,-}(x)$, $p_{-,+}(x)$, and $p_{-,-}(x)$. The combination of SLM and soft partitioning leads to a hierarchical SLM/SP tree. Each inner node denoted by m_i has its own parameters: $\boldsymbol{a_i}, \boldsymbol{a}', b_i$, and $\boldsymbol{b}'$. SLM/SP learns the parameters in the training stage. After training, SLM/SP uses them to assign an input sample to one of a set of partitioned subspaces with a path probability, as determined with Eq. 6.

3.2 SLM/SP Tree Design

Figure 2 illustrates the general structure of an SLM/SP tree. The design requires two main choices: determining the hierarchical tree structure and finding optimal parameters (i.e., $\boldsymbol{a_i}$ and b_i) at inner nodes. Each leaf node l of the SLM/SP tree has a K-D output vector, $\boldsymbol{q}^{\ell}$, whose kth element is

$$q_k^l = \frac{\exp(\varphi_k^l)}{\sum_{k=1}^{K} \exp(\varphi_k^l)}. \; k = 1, \ldots, K \quad (7)$$

where q_k^l = the probability of class k at the l-th leaf and φ_k^l = learned parameter for class k and leaf node l.

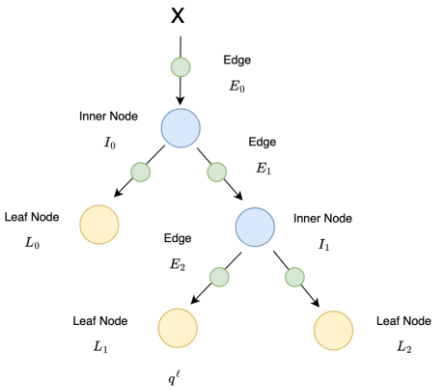

Fig. 2 Overview of an example SLM/SP tree

The two design choices are as follows:

1. *Hierarchical tree determination.* First, a greedy search finds parameters $\boldsymbol{a_i}$ and b_i at node m_i to decide if there is a gain in reducing the loss function by its splitting. If the gain is sufficient, the node is split into two. This process is applied recursively to result in a hierarchical tree. The loss function (LF) used minimizes the cross-entropy at each leaf weighted by its path probability and the target distribution. Mathematically, the LF at the l-th leaf node can be written as

$$L_l(x) = -P^l(x) \sum_k T_k^l log q_k^l \tag{8}$$

where

- $P^l(x)$ = the probability for input x to arrive at leaf node ℓ,
- q_k^l = the probability distribution at leaf node ℓ for class k, and
- T_k^l = the target distribution of class k at node ℓ obtained by putting all training samples through the tree and finding the statistics of all classes at node ℓ.

The LF evaluates the discrepancy between the predicted distribution q_k^l and the target distribution T_k^l, considering its path probability. This process greedily splits an inner node into two child nodes. We use the validation dataset to evaluate the effectiveness of SP by checking the validation accuracy improvement of the split. The split is stopped at a particular branch if there is no further improvement on the validation dataset. The maximum tree depth and the minimum sample number per node as stopping criteria are also possible. The maximum tree depth was 5 in the experiments conducted on example datasets.

2. *Node parameters determination*. After finalizing tree structure, global optimization is applied to update the PV and the bias at inner nodes for further performance improvement. The total LF from all leaf nodes can be written as

$$L = \sum_{n=1}^{N} L(x_n) = \sum_{n=1}^{N} \sum_{l \in LeafNodes} L_l(x_n) = \\ -\sum_{n=1}^{N} \sum_{l \in LeafNodes} P^l(x_n) \sum_k T_k^l log q_k^l \quad (9)$$

Since L is differentiable, we use the mini-batch stochastic gradient descent method to determine parameters $\boldsymbol{a}_i$ and b_i to minimize $L(x)$.

3.3 Inference with SLM/SP

The inference with SLM/SP can implemented in two schemes, full-path inference and single-path inference, based on a trade-off between accuracy and computation.

Full-path inference. This calculates the probabilistic distributions over all leaves. The predicted class of a single test sample, x_n, is given by

$$\hat{k}(x_n) = \arg \max_k \sum_l P^l(x_n) q_k^l(x_n) \quad (10)$$

To get the global estimation of $\hat{k}(x_n)$, we need to traverse the full SLM/SP tree to estimate the probability $P^l(x_n)$ for input x_n to arrive at leaf node l and the local probabilistic distribution $q_k^l(x_n)$ among k classes. The time and memory complexity increases exponentially with the increasing depth of SLM/SP tree, and may become too high to be practical for a large total number of test samples.

Single-path inference. For a larger test sample size, we must simplify the multi-path inference for higher efficiency of memory and time at the expense of lower accuracy. The simplified scheme adopted in our experiments is the single-path inference. It greedily traverses the tree in the directions of highest confidence of inner nodes to reach a leaf node l and then predicts its class based on the maximum likelihood at that node, namely,

$$\hat{k}(x_n) = \arg \max_k q_k^l \quad (11)$$

4. Experiments

The experiments were set up for four image classification datasets: (1) MNIST,[36] (2) Fashion-MNIST,[37] (3) CIFAR10[38] (data augmentation with random horizontal flip and random cropping was applied), and (4) STL-10[39] (images were resized from 96 × 96 to 32 × 32 pixels using bilinear interpolation). It consisted of 10 classes of images, with 5,000 images in the training set and 8,000 images in the test set.

For CIFAR-10 and STL-10 datasets, we included VGG-16[40] and LeNet-5[41] in performance benchmarking as representatives of heavy and lightweight neural networks, respectively. We followed the standard design of VGG-16 and the settings in Chen et al.[26] for LeNet-5 and adopted the same hyper-parameters in DL networks for CIFAR-10 and STL-10.

The STL-10 dataset was used to study the case of data deficiency. That was evaluated by the floating-point operations (FLOPs) in single-image inference and the model size in terms of the number of model parameters, whereas effectiveness was evaluated by classification accuracy.

Benchmarking image classification methods. For performance benchmarking, we considered some representative image classification results based on GL and DL methods. The framework is available in Figure 3. As mentioned in Section 2.1, GL-based methods have three cascaded modules: unsupervised representation learning, supervised feature learning, and supervised decision learning. Here, we compare the GL-based methods:

GL-1: pixelhop[25] for module 1 and SVM for module 3.

GL-2: pixelhop++[26] for module 1 and XGBoost for module 3. We replaced the linear classifier in Chen et al.[26] with the XGBoost Classifier, used DFT to select 2000 features for decision learning, and set the tree depth to 5 and tree number to 1000.

GL-3: pixelhop++ for module 1 and SLM/SP for module 3. We used DFT to select 512-D features as the input to SLM/SP and set the hidden layer neuron number to 512 and maximum depth of the SLM/SP tree to 5.

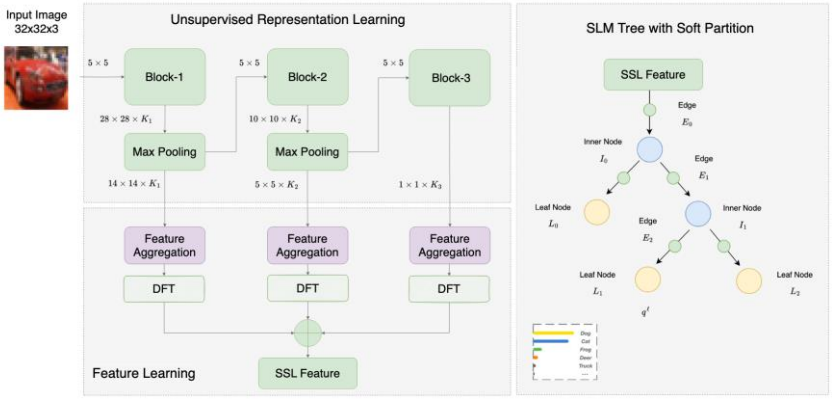

Fig. 3 Image classification framework with SLM/SP tree

5. Results and Discussion

The performance comparison of five benchmarking methods for CIFAR-10 can be categorized into lightweight and heavyweight groups depending on the inference complexity and the model size given (Table 1).

Table 1 Performance comparison for CIFAR-10

Methods	VGG-16	LeNet-5	GL-1	GL-2	GL-3 (ours)
FLOPs	875.06 M	14.74 M	21.30 M	17.73 M	20.29 M
(Against LeNet-5)	(59.40x)	(1x)	(1.45x)	(1.20x)	(1.38x)
Model size	138.36 M	395.01 K	1.66 M	739.89 K	4.39 M
(Against LeNet-5)	(350.27x)	(1x)	(4.20x)	(1.87x)	(11.11x)
Accuracy (%)	93.15	68.72	71.37	75.29	87.36

The lightweight group is comprised of LeNet-5, GL-1, GL-2, and GL-3. GL-3 uses the SLM/SP classifier, and it outperforms LeNet-5, GL-1, GL-2 by 18.64%, 15.99%, and 12.07%, respectively. GL-2 and GL-3 are almost identical, except GL-2 uses the XGBoost classifier while GL-3 uses the SLM/SP classifier. Their significant performance gap demonstrates the effectiveness of SLM/SP over XGBoost. The additional costs in inference complexity (FLOPs) and model sizes appears to be well-justified.

The heavyweight group consists of VGG-16. When compared with VGG-16, GL-3 is inferior by 5.79% in classification accuracy. However, it demands much less inference FLOPs and fewer model parameters. The savings in memory and computation are attractive for mobile and edge computing.

STL-10 was used to study the data deficiency setting, and its performance comparison is available in Table 2. Due to the small number of training data in STL-10, DL-based methods cannot benefit much from their large model sizes when compared with the CIFAR-10 dataset. GL-3 achieves the best classification

accuracy among all five benchmarking methods. At the same time, its inference complexity is close to the lowest. For the training data deficiency case, VGG-16 can only achieve classification accuracy like GL-3, but with much higher inference complexity (106x) and memory requirement (33x). The experimental results show that GL-3 can use the limited training data effectively and provide the best tradeoff between efficiency and effectiveness.

Table 2 Performance comparison for STL-10

Methods	VGG-16	LeNet-5	GL-1	GL-2	GL-3 (ours)
FLOPs	875.06 M	14.74 M	76.72 M	8.16 M	8.19 M
(Against LeNet-5)	(59.40x)	(1x)	(5.20x)	(0.55x)	(0.56x)
Model size	138.36 M	395.01 K	8.10 M	427.02 K	4.40 M
(Against LeNet-5)	(350.27x)	(1x)	(20.51x)	(1.08x)	(11.14x)
Accuracy (%)	65.75	51.89	56.48	62.07	66.15

For MNIST and Fashion MNIST datasets, we compare the performance of three benchmarking methods: LeNet-5, GL-2, and GL-3 (Tables 3 and 4). MNIST is an easy dataset. The classification accuracy saturates at 99% for most benchmarking methods. The improved classification accuracy rates of GL-3 over LeNet-5 and GL-2 are 0.26% and 0.11%, respectively. Although the gains are relatively small, they are achieved with additional inference complexity and a larger model size.

Table 3 Performance comparison for LeNet-5, GL-2, and GL-3 applied to MNIST

Methods	LeNet-5	GL-2	GL-3 (ours)
FLOPs	846.08 K	1.49 M	2.05 M
(Against LeNet-5)	(1x)	(1.76x)	(2.42x)
Model size	61.71 K	57.82 K	1.12 M
(Against LeNet-5)	(1x)	(0.94x)	(18.15x)
Accuracy (%)	99.04	99.19	99.30

Table 4 Performance comparison for Fashion LeNet-5, GL-2, and GL-3 for FMNIST

Methods	LeNet-5	GL-2	GL-3 (ours)
FLOPs	3.58 M	2.69 M	3.05 M
(Against LeNet-5)	(1x)	(0.85x)	(0.75x)
Model size	194.56 K	233.03 K	1.25 M
(Against LeNet-5)	(1x)	(1.20x)	(6.41x)
Accuracy (%)	89.74	91.37	92.20

Its classification accuracy is approximately 90% for most benchmarking methods. The improved classification accuracy rates of GL-3 over LeNet-5 and GL-2 are 2.46% and 0.83%, respectively. The inference complexity of all three methods is comparable. Although the model size of GL-3 is larger than those of LeNet-5 and

GL-2, it is still relatively small (i.e., 1.25 M). It can be accommodated well by mobile and edge devices.

6. Conclusion and Future Direction

SLM/SP is a proposed tree-based classifier that combines the input features at all inner nodes of an SLM tree, thereby differing from the classic SLM tree. The SLM/SP uses a soft DT data structure and learns by local greedy subspace partitioning. The optimal weights at each node can be obtained by optimizing a differentiable LF using the mini-batch stochastic gradient descent method. The SLM/SP classifier enables efficient training, high classification accuracy, and a small model size. It was found to be highly competitive with DL networks against easier image classification datasets. In the future, we intend to develop more powerful SLM/SP-based classification approaches for more challenging datasets, such as CIFAR-100 and ImageNet.

6. References

1. Fu H, Yang Y, Liu Y, Lin J, Harrison E, Mishra VK, Kuo C-CJ. Acceleration of subspace learning machine via particle swarm optimization and parallel processing. Proceedings of the 2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC); Chiang Mai, Thailand. IEEE; 2022. p. 1019–1024.

2. Fu H, Yang Y, Mishra VK, Kuo C-CJ. Classification via subspace learning machine (SLM): methodology and performance evaluation. Proceedings of the ICASSP 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); Rhode Island, Greece. IEEE; 2023. p. 1–5.

3. Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. Commun ACM. 2017;60(6):84–90.

4. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, et al. An image is worth 16x16 words: transformers for image recognition at scale. arXiv; 2020 [accessed 2023 Dec 1]. https://arxiv.org/abs/2010.11929.

5. Kuo C-CJ. Understanding convolutional neural networks with a mathematical model. J Vis Commun Image Represent. 2016;41:406–413.

6. Kuo C-CJ. The CNN as a guided multilayer RECOS transform [lecture notes]. IEEE Signal Processing Magazine. 2017;34(3):81–89.

7. Kuo C-CJ, Chen Y. On data-driven Saak transform. J Vis Commun Image Represent. 2018;50:237–246.

8. Kuo C-CJ, Zhang M, Li S, Duan J, Chen Y. Interpretable convolutional neural networks via feedforward design. J Vis Commun Image Represent. 2019;60:346–359.

9. Kuo C-CJ, Madni AM. Green learning: introduction, examples and outlook. J Vis Commun Image Represent. 2023;90:103685.

10. Zhang M, You H, Kadam P, Liu S, Kuo C-CJ. PointHop: an explainable machine learning method for point cloud classification. IEEE Transactions on Multimedia. 2020;22(7):1744–1755.

11. Zhang M, Wang Y, Kadam P, Liu S, Kuo C-CJ. Pointhop++: a lightweight learning model on point sets for 3D classification. Proceedings of the 2020 IEEE International Conference on Image Processing (ICIP); Abu Dhabi, UAE. IEEE; 2020. p. 3319–3323.

12. Chen H-S, Rouhsedaghat M, Ghani H, Hu S, You S, Kuo C-CJ. DefakeHop: a light-weight high-performance deepfake detector. Proceedings of the 2021 IEEE International Conference on Multimedia and Expo (ICME); Shenzhen, China. IEEE; 2021. p. 1–6.

13. Chen H-S, Hu S, Yu S, Kuo C-CJ. DefakeHop++: an enhanced lightweight deepfake detector. APSIPA Trans Signal Inf Process. 2022;11(2):e41.

14. Zhu Y, Wang X, Salloum R, Chen H-S, Kuo C-CJ. RGGID: a robust and green GAN-fake image detector. APSIPA Trans Signal Inf Process. 2022;11(2):e38.

15. Zhu Y, Wang X, Chen H-S, Salloum R, Kuo C-CJ. A-PixelHop: a green, robust and explainable fake-image detector. Proceedings of the ICASSP 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); Singapore. IEEE; 2022. p. 8947–8951.

16. Lei X, Zhao G, Kuo C-CJ. Nites: a non-parametric interpretable texture synthesis method. Proceedings of the 2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC); Auckland, New Zealand. IEEE; 2020. p. 1698–1706.

17. Lei X, Zhao G, Zhang K, Kuo C-CJ. TGHop: an explainable, efficient, and lightweight method for texture generation. APSIPA Trans Signal Inf Process. 2021;10:e17.

18. Azizi Z, Kuo C-CJ. PAGER: Progressive attribute-guided extendable robust image generation. APSIPA Trans Signal Inf Process. 2022;11(1):e35.

19. Mei Z, Wang Y-C, He X, Kuo C-CJ. GreenBIQA: a lightweight blind image quality assessment method. Proceedings of the 2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP). IEEE; 2022 Sept 26–28.

20. Liu X, Xing F, Yang C, Kuo C-CJ, Babu S, El Fakhri G, Jenkins T, Woo J. VoxelHop: successive subspace learning for ALS disease classification using structural MRI. IEEE J Biomed Health Inform. 2021;26(3):1128–1139.

21. Rouhsedaghat M, Wang Y, Ge X, Hu S, You S, Kuo C-CJ, FaceHop: a light-weight low-resolution face gender classification method. Pattern recognition: ICPR international workshops and challenges. Springer; 2021. p. 169–183.

22. Zhou Z, Fu H, You S, Borel-Donohue CC, Kuo C-CJ. UHP-SOT: an unsupervised high-performance single object tracker. 2021 International Conference on Visual Communications and Image Processing (VCIP). IEEE; 2021 Dec 5–8. p. 1–5.

23. Zhou Z, Fu H, You S, Kuo C-CJ. GUSOT: green and unsupervised single object tracking for long video sequences. Proceedings of the 2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP). IEEE; 2022 Sept 26–28.

24. Zhou Z, Fu H, You S, Kuo C-CJ. UHP-SOT++: an unsupervised lightweight single object tracker. APSIPA Trans Signal Inf Process. 2022;11(1):e27.

25. Chen Y, Kuo C-CJ. PixelHop: a successive subspace learning (SSL) method for object recognition. J Vis Commun Image Represent. 2020;70:102749.

26. Chen Y, Rouhsedaghat M, You S, Rao R, Kuo C-CJ. PixelHop++: a small successive-subspace-learning-based (SSL-based) model for image classification. Proceedings of the 2020 IEEE International Conference on Image Processing (ICIP); Abu Dhabi, UAE. IEEE; 2020. p. 3294–3298.

27. Yang Y, Fu H, Kuo C-CJ. Design of supervision-scalable learning systems: methodology and performance benchmarking. arXiv; 2022 [accessed 2023 Dec 1]. https://arxiv.org/abs/2206.09061.

28. Yang Y, Wang W, Fu H, Kuo C-CJ. On supervised feature selection from high dimensional feature spaces. APSIPA Trans Signal Inf Process. 2022;11(1):e31.

29. Jordan MI, Jacobs RA. Hierarchical mixtures of experts and the EM algorithm. Neural computation. 1994;6(2):181–214.

30. Suarez A, Lutsko JF. Globally optimal fuzzy decision trees for classification and regression. IEEE PAMI. 1999;21(12):1297–1311.

31. Léon A, Denoyer L. Policy-gradient methods for decision trees. Proceedings of the European Symposium on Artificial Neural Network; Bruges, Belgium. 2016. P. 453–458.

32. Ioannou Y, Robertson D, Zikic D, Kontschieder P, Shotton J, Brown M, Criminisi A. Decision forests, convolutional networks and the models in-between. arXiv; 2016 [accessed 2023 Dec 1]. https://arxiv.org/abs/1603.01250

33. Laptev D, Buhmann JM. Convolutional decision trees for feature learning and segmentation. Pattern Recognition: 36th German Conference, GCPR 2014; Münster, Germany. 2014 Sept 2–5. p. 95–106.

34. Kontschieder P, Fiterau M, Criminisi A, Rota Bulo S. Deep neural decision forests. Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV). IEEE; 2015 Dec 7–13. p. 1467–1475.

35. Xiao H, Xu G. Neural decision tree towards fully functional neural graph. Unmanned Systems. 2020;8(3):203–210.

36. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 1998;86(11):2278–2324.

37. Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. arXiv; 2017 [accessed 2023 Dec 1]. https://arxiv.org/abs/1708.07747.

38. Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images. University of Toronto; 2009.

39. Coates A, Ng A, Lee H. An analysis of single-layer networks in unsupervised feature learning. Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. PMLR. 2011;15:215–223.

40. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv; 2014 [accessed 2023 Dec 1]. https://arxiv.org/abs/1409.1556.

41. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD. Backpropagation applied to handwritten zip code recognition. Neural computation. 1989;1(4):541–551.

List of Symbols, Abbreviations, and Acronyms

ARL	Army Research Laboratory
DEVCOM	US Army Combat Capabilities Development Command
DFT	discriminant feature test
DL	deep learning
DT	decision tree
FLOP	floating-point operation
GL	green learning
LF	loss function
MLP	multilayer perceptron
PR	pattern recognition
PV	projection vector
SLM	subspace learning machine
SOTA	state-of-the-art
SP	soft partitioning
SVM	support-vector machine

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
1 (PDF)	DEVCOM ARL FCDD RLB CI TECH LIB
1 (PDF)	DEVCOM ARL FCDD RLA NC V MISHRA