Class Incremental Learning using Continual Backpropagation on Honey Botanic Origin Classification with Hyperspectral Imaging

Guyang Zhang
The University of Auckland
Auckland, New Zealand
gzha422@aucklanduni.ac.nz

Iman Ardekani
The University of Notre Dame Australia
Sydney, Australia
iman.ardekani@nd.edu.au

Waleed Abdulla
The University of Auckland
Auckland, New Zealand
w.abdulla@auckland.ac.nz

***Abstract*—Honey is a nutritious natural sweetener synthesized from nectar. Honey types of different botanical origins vary in quality, flavor, and health benefits, leading to distinct market values. Developing botanic origin classification techniques is crucial to protect consumers' interests. However, it is impractical to collect all the varieties of honey products at once to train a model for botanic origin differentiation. Thus, continual learning or class-incremental learning (CIL) techniques have been developed to address this challenge. This study examined multiple CIL algorithms on a real-world honey hyperspectral imaging dataset and compared their performance. A novel technique is also proposed to improve the performance of class-incremental learning algorithms by reinitializing a proportion of less-used hidden neurons to inject plasticity into neural networks, i.e., continual backpropagation (CB) algorithm. The CB method addresses the issue of loss-of-plasticity. Experiments showed that CB improved the performance of most CIL methods by 1-7%.**

***Index Terms*—Class Incremental Learning, Continual Backpropagation, Honey, Replay, Dynamic Expansion Network**

I. Introduction

Deep learning, which is usually trained on labeled data for all available classes or target groups, has advanced quickly in recent years. The current trained model cannot adjust when new data from an unseen class is added without necessitating a thorough retraining of the whole architecture. However, training data is often in stream format in the open world [1]. For example, honey, the only naturally synthesized sweetener directly consumable by humans [2], [3], has become an important commodity in the global market. The growing demand for honey products requires the authentication of honey botanical origins because the various botanical origins of honey provide distinct flavors and health benefits [4], [5]. However, it is impossible to collect all different types of honey products at once to train a model to distinguish their botanic origins. The model must incrementally learn critical information from newly introduced honey classes. The capability of a model to continuously learn over time by incorporating new knowledge while maintaining formerly obtained information is known as continual learning, lifelong learning, or incremental learning [6].

The learning technique requiring the model to be updated incrementally on new classes is called Class-Incremental Learning (CIL). One challenge of CIL is catastrophic forgetting [7], i.e., directly training the network with new classes will remove the knowledge of former ones. The other issue is loss of plasticity, i.e., the network trained on formerly known classes loses the ability to learn the new classes. Hence, effectively addressing these issues and seeking trade-offs becomes the crucial challenge for developing CIL algorithms. A good model should balance capturing new classes' characteristics and retaining the formerly learned classes' concepts. This trade-off is known as the 'stability-plasticity dilemma' in neural systems [8].

In this work, we implemented different CIL techniques on a real-world honey hyperspectral imaging dataset [9]. Hyperspectral imaging (HSI) is an advanced technique that is fast, non-destructive, and requires little sample preparation. HSI records the reflection, absorption, and emission of electromagnetic signals to capture and differentiate chemical constituent changes on the surfaces of the material [10]. Some works utilizing CIL methods on HSI datasets. For example, IL based on one-class learning (OCL) was applied to identify maize seed varieties [11]; A few-shot CIL was implemented using the Replay training strategy to categorize Chrysanthemum [12]; Knowledge distillation and replay were utilized to recognize plant species [13]; A linear programming incremental learning classifier was proposed to classify HSI remote sensing data [14]; Previous label replay strategy (PLRS) was adopted to classify remote sensing HSI data by synthesizing fused labels from previous model with pseudo-labels of the current target [15].

Moreover, we also developed a method to utilize a variation of the backpropagation algorithm, ie, continual backpropagation [16], which was designed to inject plasticity into the network to maintain variability. Continual backpropagation (CB) addresses the issue of loss of plasticity by reinitializing a small portion of less-used hidden units during incremental training [16]. To the best of our knowledge, this is the first work that examines the performance of CIL algorithms on real-world honey HSI data. In addition, this is also the first work that combines CB with other CIL techniques and examines their effects on real-world datasets.

979-8-3315-7206-8/25/$31.00

II. Materials and Methods

A. Datasets

We used a hyperspectral imaging dataset for honey products containing 25 botanical origins from New Zealand's local markets [17]. The botanical origins used for each class-incremental learning task and the number of samples for each class are shown in Table I. The hyperspectral imaging was captured under Pushbroom line-scanning mode by a SOC710VP Hyperspectral Camera from Surface Optic Corporation in California, USA, with a SchneiderKreuznach Xenoplan 35mm lens [17]. An array of halogen bulbs in a circular shape surrounded by a dome was designed as the light source. With a spectral resolution of roughly 4.9 nm, each datacube had 128 spectral bands spanning from 399.40 to 1063.79 nm. The hyperspectral image featured 12-bit data (0–4095 intensity value) for every pixel, giving it a spatial resolution of 520×696 [17]. Six acquisitions were obtained for each honey botanic origin type, and 25 samples were gathered for each acquisition. Thus, as shown in Table I, there are a total number of 8675 samples, which were normalized by the Standard Normal Variate normalization (SNV) [18] method. Details regarding the sample gathering and preprocessing techniques were provided in [19].

B. Class-Incremental Learning Methods

Among the numerous research works targeting the CIL area, we selected nine representative methods to experiment with the honey HSI dataset, including Finetune as the baseline, LwF [20], EWC [7], Replay [21], iCaRL [22], WA [23], DER [24], Foster [25], and MEMO [26]. Then, these algorithms were combined with continual backpropagation algorithm [16] to examine the effects on classification performance. These CIL methods can be grouped into categories concerning various specific focuses, which are not mutually exclusive. They were categorized according to a survey on CIL [6], whose codes of these CIL techniques were also modified to develop our work. For all these CIL algorithms, the dimension of the final fully connected layer was extended to include the number of classes seen so far.

1) The **Finetune** method, which was set as the baseline, directly used the network trained with the samples from new classes to predict the samples from all the classes seen so far. Finetune is used as the baseline method because there is no modification to the network architecture or loss function and no exemplars in the training set. It only used training samples from the current task to learn new knowledge for the current task.
2) **EWC** [7] is a Parameter Regularization method, which is based on the assumption that the contributions of network parameters are not equal. Thus, Parameter Regularization methods attempted to evaluate the importance of different parameters to the network and to maintain former knowledge by keeping the important ones static [6]. EWC addresses the parameter regularization by maintaining an importance matrix, which is estimated by Fisher information matrix.
3) **LwF** [20] utilized Knowledge Distillation (KD) [27], which enables the knowledge transfer from a teacher model to the student model, can also be deployed for CIL tasks by keeping the old network from the previous learning task and the new model updated for the current task. It balances a trade-off between the old and new networks.
4) Applying an exemplar set is an incremental learning technique, i.e., Data Replay. **Replay** can be utilized for network training by saving a small fraction of samples from previously known classes [21]. The exemplars are representative samples of each class. **iCaRL** [22] extends LwF by utilizing an exemplar selection strategy called herding, which aims to select the most representative instances. Herding calculates the center of each class and selects the samples that are more closer to the class center within the memory limit. However, despite the numerous methods to construct the exemplar set for CIL training [6], [26], [29], we did not apply any specific exemplar set selection method. We randomly selected 20 exemplars from each known class and concatenated them to the training set of the current CIL task. The efficiency and effectiveness of exemplar selection techniques are not concerns for this work.
5) There were also works attempting to explore the abnormal behaviors of networks during CIL training. UCIR [30] finds that the incrementally trained networks tend to predict samples as newly encountered classes because the weight norm of new classes is significantly larger than that of old ones. Thus, **WA** [23] normalizes the weight corresponding to the old tasks after each incremental learning task.
6) Another group of CIL techniques is Dynamic Networks [31], which are designed to dynamically expand the network's representation ability for incremental learning tasks. **DER** [24] was proposed to expand a new backbone for each new task and aggregate the features with a larger, fully connected layer. The old backbone is frozen to maintain former knowledge for new learning tasks.
7) One major drawback of DER is the infinitely growing memory size for saving networks. **FOSTER** [25] suggests that features learned by former incremental learning tasks can be compressed by knowledge distillation. The number of backbones is limited to the teacher network and student network.
8) **MEMO** [26] finds that shallow layers of networks from different CIL learning stages are more similar than deep layers, indicating that deep layers are specific to the different tasks. Hence, MEMO proposes to decouple the backbone into specialized block that corresponds to the deep layers, and generalized block that corresponds to the shallow layers. MEMO only expands specialized blocks. We used the last two fully connected layers after the last convolutional layer as the task-specific deep layers. All the convolutional layers were considered as the general feature extraction layers.

TABLE I
HONEY SAMPLE NUMBER FOR CIL TASKS

Honey Botanical Origins and Sample Sizes for Each Class-Incremental Learning Task				
Task 0	***Task 1***	***Task 2***	***Task 3***	***Task 4***
0 BBLiquid: 300	5 Field+Tawari: 150	10 ManukaUMF10: 900	15 ManukaUMF20: 275	20 Rata: 300
1 BorageField: 150	6 Honeydew: 150	11 ManukaUMF12: 150	16 ManukaUMF22: 150	21 Rewarewa: 900
2 Clover: 600	7 Kamahi: 300	12 ManukaUMF13: 150	17 ManukaUMF5: 900	22 Tawari: 450
3 Clover Cream: 150	8 Manuka: 450	13 ManukaUMF15: 600	18 Multifloral: 300	23 Wildflower: 150
4 Clover Liquid: 150	9 ManukaBlend: 450	14 ManukaUMF18: 150	19 Pohu: 300	24 Wildland: 150

C. Continual backpropagation

Continual backpropagation [16] randomly reinitializes a proportion of low-utility units in the network. The utility measure is defined to measure the magnitude and contribution of the product of units' activation and outgoing weight. If the contribution of a hidden unit is small, this hidden unit is less useful than other hidden units. The contribution utility $\mathbf{u}_l[i]$ of a ith hidden unit in layer l at time t is designed as the sum of the utilities, which is measured as a running average of contributions with a decay rate η.

$$\mathbf{u}_l[i] = \eta \times \mathbf{u}_l[i] + (1-\eta) \times |\mathbf{h}_{l,i,t}| \times \sum_{k=1}^{n_{l+1}} |\mathbf{w}_{l,i,k,t}|, \quad (1)$$

where $\mathbf{h}_{l,i,t}$ is the output of the ith hidden unit in layer l at time t, $\mathbf{w}_{l,i,k,t}$ is the weight connecting the ith unit in layer l to the kth unit in layer $l+1$ at time t, and n_{l+1} is the number of units in layer $l+1$. The outgoing weights of a reinitialized hidden unit are set to zero, ensuring that the new hidden units do not affect the function already learned. In order to prevent newly initialized hidden units, which have zero utility, from immediate reinitialization, they are protected by setting a maturity threshold m number of updates. We call a unit mature if its age is greater than m. The age of a hidden unit is related to the number of epochs, the number of batches for each learning step, and the number of hidden layers. If the maturity threshold m is set to 2000, given a network with 24 hidden layers and five mini-batches for the training dataset, the hidden units can mature every $16.67 = 2000/(24*5)$ epoch. A fraction, which is determined by the replacement rate of ρ, of mature units is reinitialized in every layer. In order to evaluate the effects of different hyperparameters on CB, we performed a sensitivity analysis on the replacement rate of ρ and maturity threshold m. In order to analyse the effects of different hyperparameters on CB, a sensitivity analysis was conducted on two essential hyperparameters, i.e. replacement rate and maturity threshold. The weighted average F1 scores on different CIL tasks are shown in the supplementary material. For each CIL method with CB, the best combination of replacement rate ρ and maturity threshold m were shown in Table II. According to the definition of the different dynamic neural networks, DER applied CB on the newly added network for current task, FOSTER utilized CB on the student network, and MEMO implemented CB on the generalized block.

TABLE II
HYPER-PARAMETER SETTING OF CB'S REPLACEMENT RATE AND MATURITY THRESHOLD ACCORDING.

Best Hyper-parameter of CB		
Method	Replacement Rate	Maturity Threshold
Finetune + CB	0.1	2000
LwF + CB	0.001	5000
EWC + CB	0.001	2000
Replay + CB	0.5	2000
iCarl + CB	0.001	5000
WA + CB	0.001	2000
DER + CB	0.1	2000
FOSTER + CB	0.5	2000
MEMO + CB	0.001	2000

III. RESULTS AND DISCUSSION

All the experiments were run on a laptop with NVIDIA Geforce GTX3060 6.0 GB GPU and 16.0GB RAM. For each sub-task, five rounds of experiments were conducted to obtain the average and standard deviation of classification scores. For each class-incremental learning task, the honey dataset used 70% for training, 15% for validation, and 15% for testing. The test results were saved based on the best validation accuracy. We started with training on five botanic origin classes and then added more, five at a time until all 25 were available. The botanic origin classes used for each incremental learning task are displayed in Table I. After each addition, the networks were trained and performance was measured for all available classes.

For all the experiments, a 1D-CNN is used as the backbone for all the CIL tasks. The parameters of kernel size / stride step / padding and output shape of each layer are shown in Table III. Adam optimizer is applied with 0.0001 learning rate. Other hyperparameters, including the λ to balance the trade-off between old and new knowledge in LwF and iCarl, utilized the default setting of experiments in [6].

A. Performance Evaluation

The weighted average F1 score was applied to evaluate the performance of the classification models. In order to ensure that the scores of the exemplar set do not significantly affect the classification outcomes, the weighted average F1-score computes the weighted mean of the F1 scores based on the number of samples of each class. The F1 score, precision, and recall scores were calculated as:

TABLE III
CNN MODEL ARCHITECTURES, THE PARAMETER COLUMN SHOW KERNEL SIZE/STRIDE STEP/PADDING, RESPECTIVELY

CNN Model Architecture		
	kernel/stride/padding	output
Layer 0	4/1/0	32x125
Layer 1	4/1/0	64x122
Layer 2	4/1/0	96x119
Layer 3	4/2/0	128x58
Layer 4	4/2/0	256x28
Layer 5	4/2/0	512x13
Layer 6	4/2/0	1024x5
Layer 7	4/2/0	2048x1
Fully Connected 256		
Fully Connected		

$$\begin{aligned} \text{F1} &= \frac{2 \times (\text{precision} \times \text{recall})}{\text{precision} + \text{recall}} \\ \text{precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}} \\ \text{recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}} \end{aligned} \tag{2}$$

True positives (TP), false positives (FP), and false negatives (FN) were used to compute these scores. Recall assesses the classifier's capacity to identify every positive sample, whereas precision gauges its capacity to avoid labeling a negative sample as positive.

B. Classification results

The classification results shown in Tables V and VI are the mean values and standard deviations of test-weighted average F1 scores from the five rounds of experiments. The model sizes and each epoch's computational time of different CIL algorithms are presented in Table IV.

1) Model sizes and computational time: Model size and computational time: In Table IV, it is evident that model sizes and each epoch's training time of dynamic network algorithms, such as DER and MEMO, significantly increase as more tasks are conducted. EWC has a higher computational cost because it has to compute a Fisher information matrix as the parameter regularization.

2) Performance of CIL methods: The performance of different CIL algorithms can be compared in Table V. Clearly, the methods using exemplars, including Replay, iCarl, WA, DER, FOSTER, and MEMO, obtain better F1 scores than Finetune, LwF, and EWC. Moreover, different methods display various performances on different CIL tasks. For instance, LwF scored higher on task 1 than EWC and Finetune, while it obtained a lower F1 score than EWC on task 2. EWC had lower F1 scores than the baseline Finetune method on tasks 3 and 4. A similar pattern can also be observed for methods with exemplars. Although DER acquired the highest scores on tasks 2 and 4, FOSTER performed better on tasks 1 and 3.

However, of all the six algorithms using exemplars, some obtained lower classification results compared to the Replay method. In Table V, iCarl, WA, and MEMO had lower F1 scores than Replay on tasks 1, 2, and 3, while they only achieved higher scores on task 4. In addition, FOSTER also had a lower F1 score than Replay on task 2, but it achieved much higher scores on tasks 1, 3, and 4.

In Fig 1, the algorithms using exemplars showed a distinct pattern when comparing performance among different tasks: unlike the three methods without exemplars, which experienced performance decline as the number of botanic origin classes increased, the algorithms using exemplars presented performance improvement from task 2 to 3. This phenomenon might be caused by the similarity of honey botanic origins between tasks 2 and 3, which mainly contain graded Mānuka honey in both tasks, as shown in Table I.

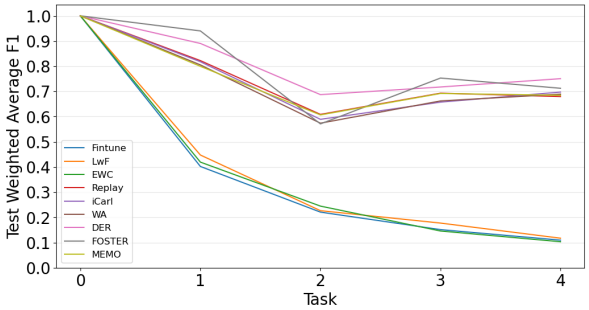

Fig. 1. Comparison of different class-incremental learning algorithms. The mean values for the test weighted average F1 scores of the five rounds of experiments were shown for each class-incremental learning task.

3) Performance of CIL methods with CB: After comparing the classification results in Tables V and VI, it seems that continuous backpropagation (CB) can be used to improve the class-incremental learning performance for most algorithms. The F1 scores of the three methods without exemplars increased after applying CB on task 4, while their performance fluctuated on tasks 1, 2, and 3: the score of Fintune + CB slightly inclined, while those of LwF + CB and EWC + CB declined on task 1; Fintune + CB and LwF + CB obtained higher F1 scores on task 2 than vanilla Fintune and LwF, while EWC + CB had a lower score with higher standard deviation; all the three algorithms with CB acquired lower F1 scores on task 3 compared to the corresponding vanilla CIL methods, while they all achieved higher F1 scores on task 4 with higher standard deviations.

After combining CB with other CIL algorithms, the performance deterioration was mainly observed by WA (on tasks 2 and 4). However, WA + CB had lower standard deviations on all the CIL tasks. The performance differences between vanilla WA and WA + CB were $\leq$2% in task 2 with $\geq$7% standard deviation and $\leq$0.1% in task 4 with $\geq$3% standard deviation. These deviations might not be significant, given the large standard deviation of their F1 scores.

Although DER, FOSTER, and MEMO are dynamic networks, FOSTER uses model compression to restrain the growth of model sizes, DER expands a new backbone for each new task, and MEMO uses the same generalized block for all tasks. The reinitialization of neurons was more effective for

TABLE IV
INCREMENTAL LEARNING MODEL SIZE AND COMPUTATIONAL COST. IN EACH CELL, THE MODEL SIZE, EACH EPOCHS'S TRAINING TIME FOR VANILLA CIL METHOD AND CB METHOD ARE SHOWN.

CIL	Model Sizes and Each Epoch's Training Time of Incremental Task No.				
Method	***0***	***1***	***2***	***3***	***4***
Finetune	46MB, 0.11s, cb:0.11s	46MB, 0.13s, cb:0.13s	46MB, 0.15s, cb:0.15s	46MB, 0.15s, cb:0.15s	46MB, 0.16s, cb:0.16s
LwF	46MB, 0.11s, cb:0.11s	46MB, 0.16s, cb:0.16s	46MB, 0.19s, cb:0.19s	46MB, 0.19s, cb:0.19s	46MB, 0.20s, cb:0.20s
EWC	46MB, 0.23s, cb:0.23s	46MB, 0.37s, cb:0.37s	46MB, 0.38s, cb:0.38s	46MB, 0.42s, cb:0.42s	46MB, 0.43s, cb:0.43s
Replay	46MB, 0.10s cb:0.10s	46MB, 0.13s, cb:0.13s	46MB, 0.18s, cb:0.18s	46MB, 0.17s, cb:0.17s	46MB, 0.18s, cb:0.18s
iCarl	46MB, 0.10s, cb:0.10s	46MB, 0.17s, cb:0.17s	46MB, 0.22s, cb:0.22s	46MB, 0.23s, cb:0.23s	46MB, 0.23s, cb:0.23s
WA	46MB, 0.11s, cb:0.11s	46MB, 0.16s, cb:0.16s	46MB, 0.22s, cb:0.22s	46MB, 0.23s, cb:0.23s	46MB, 0.23s, cb:0.23s
DER	46MB, 0.11s, cb:0.11s	92MB, 0.19s, cb:0.19s	138MB, 0.29s, cb:0.29s	184MB, 0.34s, cb:0.34s	230MB, 0.38s, cb:0.38s
FOSTER	46MB, 0.11s, cb:0.11s	92MB, 0.20s, cb:0.20s	92MB, 0.27s, cb:0.27s	92MB, 0.28s, cb:0.28s	92MB, 0.30s, cb:0.30s
MEMO	46MB, 0.11s, cb:0.11s	80MB, 0.19s, cb:0.19s	115MB, 0.31s, cb:0.31s	150MB, 0.37s, cb:0.37s	185MB, 0.43s, cb:0.43s

TABLE V
INCREMENTAL LEARNING CLASSIFICATION RESULTS

CIL	Weighted Average F1 Score of Incremental Task No.				
Method	***0***	***1***	***2***	***3***	***4***
Retrain	100.00 ± 0.00%	100.00 ± 0.00%	100.00 ± 0.00%	99.78 ± 0.08%	99.83 ± 0.19%
Finetune	100.00 ± 0.00%	40.20 ± 7.26%	22.07 ± 3.44%	15.12 ± 2.00%	10.94 ± 1.14%
LwF	100.00 ± 0.00%	44.74 ± 8.84%	22.64 ± 5.70%	17.72 ± 3.57%	11.71 ± 1.35%
EWC	100.00 ± 0.00%	41.92 ± 4.16%	24.46 ± 2.80%	14.57 ± 2.07%	10.31 ± 1.16%
Replay	100.00 ± 0.00%	82.22 ± 5.26%	60.87 ± 6.71%	69.29 ± 4.06%	67.94 ± 4.98%
iCarl	100.00 ± 0.00%	81.78 ± 2.15%	58.91 ± 7.20%	65.73 ± 4.14%	69.74 ± 4.33%
WA	100.00 ± 0.00%	80.52 ± 4.05%	57.38 ± 8.38%	66.21 ± 6.13%	68.91 ± 6.56%
DER	100.00 ± 0.00%	89.04 ± 4.88%	68.71 ± 4.61%	71.75 ± 2.55%	75.05 ± 4.28%
FOSTER	100.00 ± 0.00%	94.01 ± 2.63%	57.09 ± 7.49%	75.30 ± 4.69%	71.24 ± 2.45%
MEMO	100.00 ± 0.00%	76.06 ± 5.32%	54.81 ± 7.14%	65.05 ± 7.20%	63.51 ± 7.37%

FOSTER, which contains only two sub-networks, than DER, which includes one network backbone for each CIL task.

TABLE VI
INCREMENTAL LEARNING WITH CONTINUAL BACKPROPAGATION CLASSIFICATION RESULTS

CIL	Weighted Average F1 Scores of Incremental Task No.				
Method	***0***	***1***	***2***	***3***	***4***
Finetune+CB	100.00 ± 0.00%	42.06 ± 4.33%	26.70 ± 2.10%	14.41 ± 1.78%	11.37 ± 1.82%
LwF+CB	100.00 ± 0.00%	43.25 ± 6.92%	23.49 ± 4.33%	14.58 ± 2.30%	13.40 ± 1.53%
EWC+CB	100.00 ± 0.00%	41.81 ± 6.36%	23.99 ± 3.97%	14.01 ± 1.03%	10.90 ± 2.67%
Replay+CB	100.00 ± 0.00%	83.86 ± 4.81%	66.18 ± 6.03%	73.20 ± 2.04%	70.11 ± 2.13%
iCarl+CB	100.00 ± 0.00%	83.24 ± 1.67%	55.35 ± 3.86%	68.13 ± 1.98%	71.89 ± 1.68%
WA+CB	100.00 ± 0.00%	81.58 ± 3.19%	55.53 ± 7.75%	69.00 ± 2.83%	68.84 ± 3.42%
DER+CB	100.00 ± 0.00%	86.64 ± 2.90%	72.50 ± 1.85%	73.13 ± 2.08%	75.73 ± 5.48%
FOSTER+CB	100.00 ± 0.00%	97.20 ± 0.99%	68.87 ± 6.33%	79.20 ± 3.96%	74.34 ± 3.51%
MEMO+CB	100.00 ± 0.00%	82.31 ± 2.94%	60.42 ± 5.41%	71.54 ± 1.51%	70.94 ± 3.13%

4) Confusion Matrix: Figs. 3 and 4 display the confusion matrix of DER and FOSTER's classification results on task 4 from a randomly chosen experiment round. In Fig. 3, it is clear that DER tends to misclassify Mānuka, MānukaUMF5, and

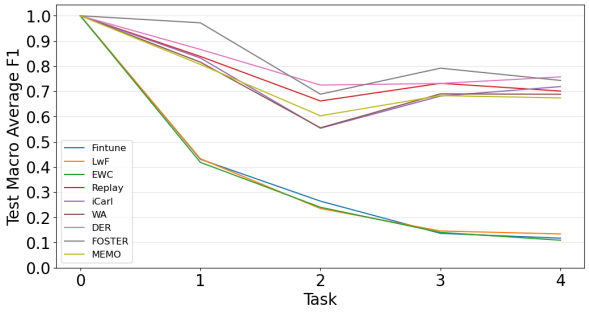

Fig. 2. Comparison of different class-incremental learning algorithms with continual backpropagation. The mean values for the test weighted average F1 scores of the five rounds of experiments were shown for each class-incremental learning task.

MānukaUMF10 as Rewarewa honey. In addition, DER also confused some MānukaUMF10 samples with MānukaUMF5 and Clover honey samples with Tawari. In Fig. 4, FOSTER also misclassifies some MānukaUMF10 samples as MānukaUMF5 and a few MānukaUMF15 / MānukaUMF18 samples as other UMF graded Mānuka honey types. There are also ungraded honey samples of Mānuka, MānukaUMF10, and MānukaUMF12 are confused with Rewarewa honey. Some Clover honey samples are falsely identified as Pohu and Tawari honey as well. We can tell from Figs. 3 and 4 that the incrementally trained models are inclined to misclassify Mānuka and UMF graded Mānuka honey as other UMF graded Mānuka honey and Rewarewa honey, while Clover honey can also be confused with Tawari honey.

IV. CONCLUSION

This paper aims to apply the honey hyperspectral imaging dataset to the class-incremental learning tasks. A variety of CIL algorithms were examined using the honey HSI dataset. Moreover, we proposed to combine the continuous backpropagation algorithm with other CIL methods. Most algorithms experienced performance improvement after applying CB, as shown in Tables V and VI. The neural networks in CIL tasks can attain more plasticity to improve the classification performance by reinitializing a small fraction of less-used neurons.

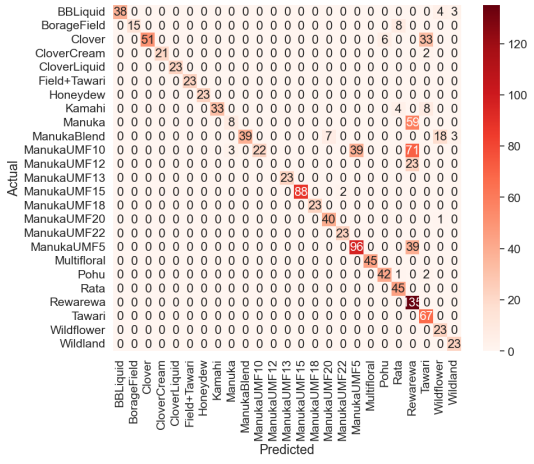

Fig. 3. Confusion matrix of DER on Task 4.

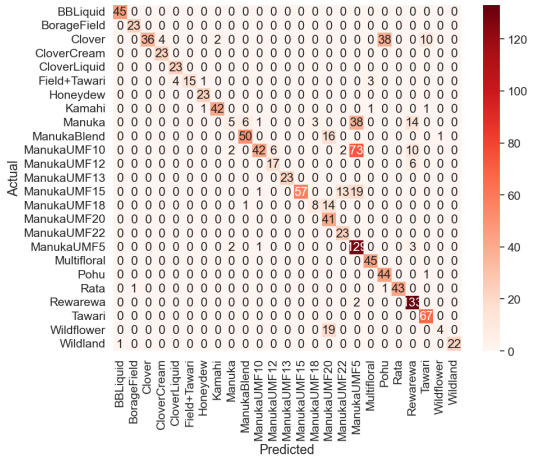

Fig. 4. Confusion matrix of FOSTER on Task 4.

REFERENCES

[1] M. De Lange et al., "A Continual Learning Survey: Defying Forgetting in Classification Tasks", *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 44, no. 7, pp. 3366–3385, 2022.

[2] J. F. H. O. Codex Alimentarius, Revised Codex Standard for Honey CODEX STAN 12-1981 Rev. 1 (1987). *World Health Organization: Food and Agriculture Organization of the United Nations*, 1987.

[3] European Commission, "European Commission Council Directive 2001/110/EC of 20 December 2001 relating to honey", *Official Journal of the European Communities*, vol. L 010, p. 12.1.2002, 47–52, 2001.

[4] F. Ulberth, "26 - Advances in Testing for Adulteration in Honey", in *Advances in Food Authenticity Testing*, G. Downey, Ed. Woodhead Publishing, 2016, pp. 729–753.

[5] A. J. Siddiqui, S. G. Musharraf, M. I. Choudhary, and Atta-ur-Rahman, "Application of analytical methods in authentication and adulteration of honey", *Food Chem.*, vol. 217, pp. 687–698, 2017.

[6] D.-W. Zhou, Q.-W. Wang, Z.-H. Qi, H.-J. Ye, D.-C. Zhan, and Z. Liu, "Class-Incremental Learning: A Survey", *IEEE Trans. Pattern Anal. Mach. Intell*, pp. 1–20, 2024.

[7] J. Kirkpatrick et al., "Overcoming catastrophic forgetting in neural networks", *Proc. of the Nat. Acad. of Sci.*, vol. 114, pp. 3521–3526, 2016.

[8] L. Wang, X. Zhang, H. Su, and J. Zhu, "A Comprehensive Survey of Continual Learning: Theory, Method and Application", *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 46, no. 8, pp. 5362–5383, 2024.

[9] A. Noviyanto and W. H. Abdulla, "Honey botanical origin classification using hyperspectral imaging and machine learning", *J. Food Eng.*, vol. 265, p. 109684, 2020.

[10] G. ElMasry and D.-W. Sun, "CHAPTER 1 - Principles of Hyperspectral Imaging Technology", in *Hyperspectral Imaging for Food Quality Analysis and Control*, D.-W. Sun, Ed. San Diego: Academic Press, 2010, pp. 3–43.

[11] L. Zhang, D. Wang, J. Liu, and D. An, "Vis-NIR hyperspectral imaging combined with incremental learning for open world maize seed varieties identification", *Comput. and Electron. in Agr.*, vol. 199, p. 107153, 2022.

[12] Z. Cai et al., "Identification of chrysanthemum using hyperspectral imaging based on few-shot class incremental learning", *Comput. and Electron. in Agr.*, vol. 215, p. 108371, 2023.

[13] H. -Y. Chien, K. -H. Liu and C. Lin, "Plant Species Recognition Based on Continual Learning Strategy," *IGARSS 2024 - 2024 IEEE Int. Geosci. Remote Sens. Symp.*, Athens, Greece, 2024, pp. 10363-10367.

[14] J. Bai et al., "Class Incremental Learning With Few-Shots Based on Linear Programming for Hyperspectral Image Classification," in *IEEE Trans. on Cybern.*, vol. 52, no. 6, pp. 5474-5485, June 2022.

[15] W. Dong, Y. Chen, S. Xiao, J. Qu and Y. Li, "Incremental Detection of Hyperspectral Targets in Consistent Scenes With Continuous Learning," in *IEEE Trans. on Geosci. Remote Sens.*, vol. 62, pp. 1-13, 2024, Art no. 5531513.

[16] S. Dohare, J. F. Hernandez-Garcia, Q. Lan, P. Rahman, A. R. Mahmood, and R. S. Sutton, "Loss of plasticity in deep continual learning", *Nature*, vol. 632, no. 8026, pp. 768–774, Aug. 2024.

[17] A. Noviyanto and W. H. Abdulla, "Segmentation and calibration of hyperspectral imaging for honey analysis", *Comput. and Electron. in Agr.*, vol. 159, pp. 129–139, 2019.

[18] R. J. Barnes, M. S. Dhanoa, and S. J. Lister, "Standard Normal Variate Transformation and De-trending of Near-Infrared Diffuse Reflectance Spectra", *Appl. Spectrosc.*, vol. 43, no. 5, pp. 772–777, May 1989.

[19] G. Zhang and W. Abdulla, "Optimizing Hyperspectral Imaging Classification Performance with CNN and Batch Normalization", *Appl. Spectrosc. Pract.*, vol. 1, no. 2, p. 27551857231204622, 2023.

[20] Z. Li and D. Hoiem, "Learning Without Forgetting", in *European Conf. on Comp. Vis. ECCV 2016*, 2016, pp. 614–629.

[21] R. Ratcliff, "Connectionist models of recognition memory: constraints imposed by learning and forgetting functions", *Psychol. Rev.*, vol. 97 2, pp. 285–308, 1990.

[22] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental Classifier and Representation Learning", in *2017 IEEE/CVF Conf. on Comp. Vis. and Pattern Recognit. (CVPR)*, 2017, pp. 5533–5542.

[23] B. Zhao, X. Xiao, G. Gan, B. Zhang, and S.-T. Xia, "Maintaining Discrimination and Fairness in Class Incremental Learning", in *2020 IEEE/CVF Conf. on Comp. Vis. and Pattern Recognit. (CVPR)*, 2020, pp. 13205–13214.

[24] S. Yan, J. Xie, and X. He, "DER: Dynamically Expandable Representation for Class Incremental Learning", in *2021 IEEE/CVF Conf. on Comp. Vis. and Pattern Recognit. (CVPR)*, 2021, pp. 3013–3022.

[25] F.-Y. Wang, D.-W. Zhou, H.-J. Ye, and D.-C. Zhan, "FOSTER: Feature Boosting and Compression for Class-Incremental Learning", in *European Conf. on Comp. Vis. ECCV 2022*, 2022, pp. 398–414.

[26] D.-W. Zhou, Q.-W. Wang, H.-J. Ye, and D.-C. Zhan, "A Model or 603 Exemplars: Towards Memory-Efficient Class-Incremental Learning", in *Int. Conf. on Learn. Rep. (ICLR)*, 2023.

[27] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network", *ArXiv*, vol. abs/1503.02531, 2015.

[28] D. N. Barry and B. C. Love, "A neural network account of memory replay and knowledge consolidation", *Cerebral Cortex*, vol. 33, no. 1, pp. 83–95, 02 2022.

[29] Y. Liu, B. Schiele, and Q. Sun, "RMM: Reinforced Memory Management for Class-Incremental Learning", in *Adv. in Neural Inform. Process. Syst.*, 2021.

[30] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a Unified Classifier Incrementally via Rebalancing", in *2019 IEEE/CVF Conf. on Comp. Vis. and Pattern Recognit. (CVPR)*, 2019, pp. 831–839.

[31] J. Yoon, E. Yang, J. Lee, and S. J. Hwang, "Lifelong Learning with Dynamically Expandable Networks", in *Int. Conf. on Learn. Rep. (ICLR)*, 2018.