2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)
August 20-24, 2022. Mexico City, Mexico

2022 IEEE 18th International Conference on Automation Science and Engineering (CASE) | 978-1-6654-9042-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/CASE49997.2022.9926567

Improved Representations for Continual Learning of Novel Motor Health Conditions through Few-Shot Prototypical Networks

Matthew Russell* and Peng Wang

Abstract— **Intelligent machine condition monitoring (CM) for automatic fault diagnosis relies on data-driven algorithms to characterize machine health for predictive maintenance activities on smart factory floors. Since data collection can be expensive, CM data sets may not cover all the possible fault conditions, necessitating that CM algorithms continually learn new conditions. State-of-the-art CM research has focused on detecting unknown conditions rather than integrating unknown conditions into future predictions. Therefore, CM-ready Continual Learning (CL) solutions should learn to classify new conditions and use improved representations that minimize the need for future fine-tuning. Meta-learning approaches like Few-Shot Prototypical Networks (FSPN) regularize base-task learning to find these more generalizable representations. Experiments on a motor data set demonstrate that FSPN with only 5 or 10 examples of the novel fault consistently outperforms static, fine-tuning, and Elastic Weight Consolidation (EWC) approaches for CL, increasing the overall accuracy by up to 19 points (53% to 72%). Compared to recent FSPN work for image classification, these results show that FSPN may be advantageous for CM due to the limited class diversity of CM data sets. Future work should extend the FSPN architecture to include open set recognition and quantitatively analyze varying numbers of base-task classes.**

Index Terms— **Continual Learning, Failure Detection and Recovery, Deep Learning Methods**

I. Introduction

Smart factories depend on intelligent machine condition monitoring (CM) to assess machine health and schedule predictive maintenance. However, extensive data from diverse operating conditions are typically not available to train data-driven CM algorithms due to cost, time, and operational constraints. Thus, CM algorithms cannot assume that the distribution of training data covers the full range of possible conditions [1], [2]. As a result, practical machine learning (ML) approaches to CM must identify novel fault conditions as they occur and distill the information needed to recognize future recurrences.

The broader ML community refers to this problem as continual (or incremental, lifelong, etc.) learning. Given a set of tasks seen one at a time, an effective Continual Learning (CL) algorithm will learn a new task without losing performance on previously seen tasks. Sometimes previous tasks may be revisited, but this study focuses on tasks seen sequentially without repetition, since this best reflects the CM application scenario in which previous fault conditions may not recur with any regularity. In addition, task boundaries are unknown in CM, making it an unsupervised CL problem comprised of two components: 1) automatic detection of new tasks (e.g., new fault conditions) and 2) continual learning of new tasks without catastrophic forgetting of previous tasks.

Related CM research has focused on detecting unknown faults—the first subproblem of unsupervised CL. Also known as open set recognition [3], detecting unknown faults requires detecting when the input data meaningfully deviates from the original training data. A popular approach is to model the probability distribution of features from each known class and use a likelihood threshold to flag new data as out of domain. Zhang et al. [1] adopted this idea by fitting a Gaussian distribution to the features extracted from gearbox vibration data under different conditions. However, similar conditions confused the approach, and the study only considered the appearance of a single novel fault. In a similar vein, Yu et al. [2] adapted work by [3], leveraged a Weibull distribution to reject unknown bearing faults, but did not incorporate these new faults into the network's future predictions. Following [4], Li et al. [5] demonstrated novel fault detection during transfer learning, and [6] extended the idea to multiple novel faults but required known and unknown fault examples to be available simultaneously, an additional training phase, and multiple clustering trials. Furthermore, none of these studies integrate emerging faults into future predictions and therefore cannot be considered CL.

Mainstream ML and deep learning (DL) research into CL has generated several methods for mitigating catastrophic forgetting. Teaching a new task to a model is trivial (e.g., through retraining or fine-tuning); remembering previous tasks during this process is extraordinarily difficult. Learning without Forgetting (LwF) [7] demonstrated how to preserve previous task information when learning a new task by discouraging weight updates that change the input-output relationship of previous task prediction layers. Nearly concurrently, Incremental Classifier and Representation Learning (iCaRL) [8] sought to mitigate catastrophic forgetting with a combination of knowledge distillation [9] from historical instances of the network and a set of class exemplars that the network can review to refresh memory of previous tasks. Avoiding the need for exemplars, Elastic Weight Consolidation (EWC) [10] developed a regularization term grounded in information theory that blocks changes to weights strongly tied to good classification performance on a previous task

This work is supported by the National Science Foundation under Grant No. 2015889

*Corresponding author: Matthew Russell (phone: 1-859-317-3373; e-mail: matthew.russell@uky.edu)

Matthew Russell is with the Department of Electrical and Computer Engineering, University of Kentucky, Lexington, KY 40506 USA (e-mail: matthew.russell@uky.edu).

Peng Wang is with the Department of Electrical and Computer Engineering and the Department of Mechanical Engineering, University of Kentucky, Lexington, KY 40506 USA (e-mail: edward.wang@uky.edu).

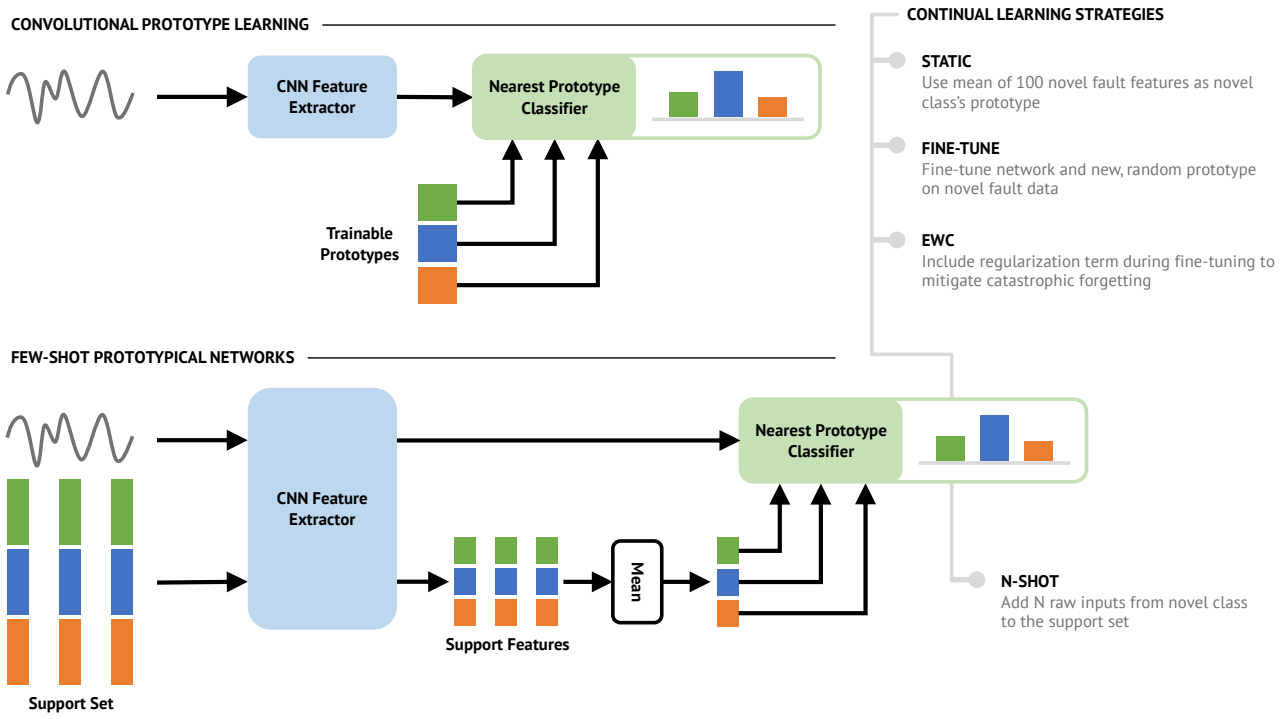

Fig. 1. Convolutional Prototype Learning (CPL) and Few-Shot Prototypical Networks (FSPN) adopt different strategies for learning good feature extraction. CPN relies on a set of trainable prototypes, while FSPN dynamically calculates prototypes from a support set as needed. This difference affects the generalizability of representations for continual learning.

and relies on the remaining network capacity to capture the new task. Hyperparameters control the tradeoff between learning new information and remembering old information. Indicative of growing interest in CL for manufacturing, Tercan et al. [11] adopted a technique similar to EWC called Memory Aware Synapses (MAS) [12] to learn successive product quality prediction tasks and eliminate the need to store exemplars from previous tasks. Intuitively, however, CM hardware is more likely to lack computational resources than sufficient memory since exemplar sets are usually small, and devices must consume as little power as possible. Therefore, practitioners cannot transfer mainstream ML work to CM without addressing the dependence on fine-tuning.

Many existing approaches rely on extensive fine-tuning because conventional CL thought assumes that task information is stored implicitly in the network weights Fine-tuning is necessary to update these weights with information from new tasks while ideally preserving old information, possibly by incorporating buffers of previous task exemplars. To alleviate fine-tuning requirements, the model should not learn the task information itself, but *how to extract relevant, discriminative information* when given examples of the tasks to complete. That is, instead of learning to answer a question like, "What shape is this example?", the model should learn to answer the question, "Given these possible shapes, what shape is most like this example?" The network learns not to extract the shape itself from the input, but the *relevant information for distinguishing shapes*. That is, rather than attempting direct shape classification, the network may learn to count sides or corners, a strategy that will better generalize to never-before-seen shapes.

Coincidentally, Few-Shot Learning (FSL) reifies this exact meta-learning objective. In K-way, n-shot FSL, the network is provided a support set with n examples for each of K classes and asked to classify a query example according to this information. Prototypical networks [13] have emerged as a straightforward but effective FSL technique and learn an embedding in which the extracted feature of the query example will be nearest to the mean feature of the n same-class elements from the support set. While the desired inter-class separation and intra-class compactness goals can be added to traditional training objectives [14], prototypical networks "train like they test" by training with few-shot episodes mirroring those seen at test time. This ensures that the network targets the true objective instead of a proxy. Gidaris et al. [15] have shown how to hone prototypical networks for CL scenarios such as Few-Shot Class-Incremental Learning (FSCIL), but the scope of this work focuses on evaluating the inherent reusability (i.e., generalizability) of features from few-shot prototypical networks versus traditional classifiers, classifiers with fine-tuning, and classifiers using EWC to mitigate forgetting.

Given the perceived theoretical meta-learning advantages of FSL for CL, this study proposes that representations learned by a Few-Shot Prototypical Network (FSPN) provide a more effective starting point for CL in CM applications

than traditional classifier-trained embeddings or state-of-the-art CL methods like EWC. This paper contributes:

1) the observation that CM applications impose unique constraints on CL, including fine-tuning restrictions on low-power edge devices, that preclude direct usage of state-of-the-art CL algorithms;
2) experimental results confirming that representations learned by FSPN better differentiate novel motor faults than representations found by a traditional classifier, even when fine-tuning is permitted (with and without EWC); and
3) analysis indicating that this advantage of few-shot representations for CM could be related to the low number of unique classes in CM data sets.

The remainder of this paper contains background on the evaluated methods in Section II, an overview of the experimental setup in Section III, presentation and discussion of the results in Section IV, and concluding thoughts and future directions in Section V.

II. Continual Learning Strategies

Continual Learning (CL) can be modeled as an ordered sequence of T tasks $\{\{\mathcal{D}^t_{\text{train}}, \mathcal{D}^t_{\text{test}}\}\}_{t=1}^{T}$ where each task has a training dataset $\mathcal{D}^t_{\text{train}}$ (that could be further split into a validation set) and a testing dataset $\mathcal{D}^t_{\text{test}}$. CL attempts to learn subsequent tasks without losing performance on preceding tasks [16]. The first, or base, task may be considered a pretraining stage without the same constraints (e.g., few-shot or no fine-tuning) as future tasks [16]. For example, a classifier might be trained on data from normal operating conditions and limited faulty conditions (Task 1) and then asked to learn one or more never-before-seen faults (future tasks) without losing the ability to classify those previously seen. The following sections introduce Convolutional Prototype Networks (CPN), Few-Shot Prototypical Networks, and Elastic Weight Consolidation (EWC) and how they can be used for CL.

A. Convolutional Prototype Networks

Convolutional Prototype Networks (CPN) [14] serve as the baseline prototypical network architecture. CPN uses a Convolutional Neural Network (CNN) for feature extraction followed by a nearest-prototype classifier using Euclidean distance. That is, given parameterized feature extraction CNN f_θ and a set of prototypes $P = \{\mathbf{p}_k \mid 1 \leq k \leq K\}$ for K classes, the CPN output is

$$p(y = k \mid \mathbf{x}, P, \theta) = \frac{-d(f_\theta(\mathbf{x}), \mathbf{p}_k)}{\sum_j -d(f_\theta(\mathbf{x}), \mathbf{p}_j)} \tag{1}$$

where $d(\cdot, \cdot)$ is Euclidean distance, and $p(y = k \mid \mathbf{x}, P, \theta)$ is the probability that example $\mathbf{x}$ belongs to class k based on prototypes P and parameters θ. Calculating the Negative Log Likelihood (NLL) of (1) gives the loss function for training CPN (this study excludes the supplementary loss terms introduced in [14]):

$$J(\mathbf{x}, P, \theta) = d(f_\theta(\mathbf{x}), \mathbf{p}_k) + \log \sum_j -d(f_\theta(\mathbf{x}), \mathbf{p}_j) \tag{2}$$

To find the feature extraction parameters θ and the class prototypes P, the resulting optimization problem can be written as

$$\left\{\hat{\theta}, \hat{P}\right\} = \underset{\{\theta, P\}}{\arg\min}\ \mathrm{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[J(\mathbf{x}, P, \theta)\right] \tag{3}$$

and seeks the joint values of θ and P that minimize the expected loss value across examples from training distribution $p(\mathbf{x})$.

For CL, CPN is trained on an initial task containing base classes $k = 1, 2, \ldots, K$. Following the idea of weight imprinting [17], the model can be extended to recognize a new class without fine-tuning by embedding n examples of the novel class into the feature space and using the mean feature as the new class prototype:

$$\mathbf{p}_{K+1} = \frac{1}{n}\sum_{i=1}^{n} f_\theta\left(\mathbf{x}^{(i)}_{K+1}\right) \tag{4}$$

$$P_{K+1} = P \cup \{p_{K+1}\} \tag{5}$$

where $\mathbf{x}^{(i)}_{K+1}$ is the ith example of novel class $K+1$, and P_{K+1} is the updated set of prototypes. Since Euclidean distance is a Bregman divergence/distance, $\mathbf{p}_{K+1}$ minimizes its expected distance to the novel class examples [13]. Once the model calculates the new prototype, it can predict the probability of the novel class for future examples using (1).

B. Elastic Weight Consolidation

While many CM applications do not support fine-tuning, a thorough analysis should still compare fine-tuning with FSPN to best understand the tradeoffs. Naive fine-tuning causes catastrophic forgetting, but Elastic Weight Consolidation (EWC) [10] mitigates this by discouraging updates to weights valuable to previous task t with a regularization term when training on task $t+1$:

$$\mathcal{L}_{\text{EWC},t+1} = \lambda \sum_i F_{i,t}\left(\theta_i - \theta_{i,t}\right)^2 \tag{6}$$

where θ_i is parameter i in the current model, $\theta_{i,t}$ is the value of θ_i after learning previous task j, and $F_{i,t}$ is the value in task t's Fisher Information Matrix (FIM) diagonal corresponding to $\theta_{i,t}$. $F_{i,t}$ represents the amount of information in $\theta_{i,t}$ about task t. Larger $F_{i,j}$ indicates that $\theta_{i,t}$ changes rapidly around its local minimum and that moving θ_i far from $\theta_{i,t}$ would significantly impact the network performance. The regularization term thus penalizes significant modifications to "important" weights from task t.

EWC requires computing and saving separate FIMs for each successive task $t = 1, 2, \ldots$, which may not be desirable if the model should learn an indefinite number of future tasks. Progress & Compress [18] introduced online EWC that combines successive FIMs with a discount rate γ:

$$F_{i,t+1} = F_i + \gamma F_{i,t} \tag{7}$$

where F_i is the FIM diagonal value for θ_i from task $t+1$, and $F_{i,t+1}$ is the updated FIM for regularizing future tasks according to (6). For CPN, fine-tuning with online EWC can supplement the CL steps described in Section II-A to

optimize f_θ for the new task. By including EWC regularization, fine-tuning should improve performance on the new task and avoid destructively manipulating weights valuable to previous tasks.

C. Few-Shot Prototypical Network

Since CPN jointly trains both the feature extractor and prototypes, it moves each class prototype towards the center of the corresponding feature cluster while simultaneously moving each feature toward its class prototype. The randomly-initialized prototypes influence feature learning without being grounded in relevant task information from the data set. This physically-disconnected randomness means that the network cannot be guaranteed to cluster future faults based on physical signal similarity. Few-Shot Prototypical Networks (FSPN) remedy this by computing prototypes from a few randomly-sampled examples of each class (the support set) [13]. The computed prototypes stem directly from features extracted from the physical signal, providing more reliable feedback for physically-meaningful clustering. Resampling the support set throughout training ensures that a variety of class examples influence the learned representation. With this revised training procedure, an FSPN only needs a few shots (examples) of each class to make accurate predictions at inference time.

Formally, an FSPN has a support set S that contains n examples for each of K classes:

$$S = \left\{ \mathbf{x}_k^{(i)} \sim p(\mathbf{x} \mid y = k) \mid 1 \leq i \leq n, 1 \leq k \leq K \right\} \quad (8)$$

where $\mathbf{x}_k^{(i)}$ is the ith example of the kth class. Instead of a single set of trainable prototypes P like CPN, the model recomputes prototypes for each training step (episode) using only the randomly sampled support set:

$$P(S) = \left\{ \frac{1}{n} \sum_{i=1}^{n} f_\theta \left(\mathbf{x}_k^{(i)} \right) \middle| 1 \leq k \leq K \right\} \quad (9)$$

The resulting few-shot classifier training problem is

$$\hat{\theta} = \arg\min_\theta \mathrm{E}_{\mathbf{x} \sim p(\mathbf{x}), S \sim p(S)} \left[J(\mathbf{x}, \theta, P(S)) \right] \quad (10)$$

Given support set S at test time, the model computes the prototypes according to (9) and class probabilities using (1). As with CPN, the model is pretrained on a base task. When n examples of novel class $K+1$ appear, they are added to the support set. The model will then classify future examples into the $K+1$ classes without fine-tuning by recomputing the prototypes using (9) and applying (1).

III. Experiments

To evaluate the proposed methods, a SpectraQuest Machinery Fault Simulator (MFS) Magnum simulated eight motor operating conditions: normal, faulted bearings, bowed rotor, broken rotor, misaligned rotor, unbalanced rotor, phase loss, and unbalanced voltage. The MFS was run at 2000 RPM and 3000 RPM, and a magnetic brake added loads of either 0.06 N·m and 0.7 N·m (load settings of 0 and 3). Each of the 32 combinations of condition and process parameters ran for 60 s in steady-state while the data acquisition software sampled vibration data at 12 kHz from a vertically-aligned and magnetically-mounted accelerometer. Data preprocessing steps normalized the signals to $[-1, 1]$ and split them into non-overlapping, 512-sample windows.

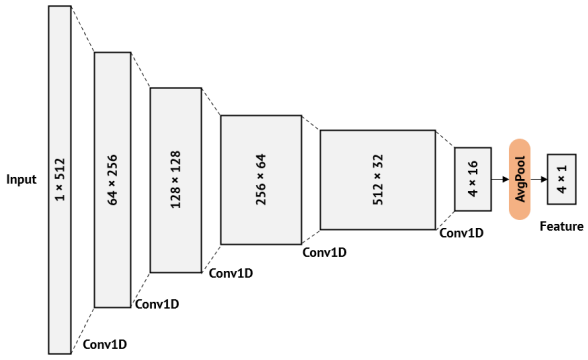

Fig. 2. Architecture of the 1D CNN for extracting latent features from motor vibration signals. A stride of 2 halves the spatial length after each convolution, all of which are followed by ReLU and batch normalization. The label $C \times L$ indicates a layer with C channels and spatial length L.

TABLE I
Task Epics

	Epic 1	Epic 2	Epic 3
Task 1	normal faulted bearings bowed rotor phase loss	normal faulted bearings bowed rotor broken rotor	normal bowed rotor broken rotor phase loss
Task 2	broken rotor	faulted bearings	faulted bearings
Task 3	misaligned rotor	phase loss	misaligned rotor
Task 4	unbal. rotor	misaligned rotor	unbal. rotor
Task 5	unbal. voltage	unbal. voltage	unbal. voltage

The eight condition classes are partitioned into five tasks for CL, and each task is split into 70% train, 15% validation, and 15% test subsets. The first task contains four conditions and always includes the normal condition, while the remaining four tasks incrementally add additional faults. Since the exact division of conditions into tasks could affect the results, three different splits (or "epics") constitute three unique testing scenarios (see Table I).

The experiments evaluate CPN without fine-tuning ("Static"), CPN with fine-tuning ("Fine Tune"), and CPN with fine-tuning plus EWC ("EWC"), along with 1-Shot, 5-Shot, and 10-Shot prototypical networks. Experiments have two phases:

Phase 1: Pretrain five sets of 1D CNN-based CPN and n-shot networks on vibration data from Task 1 with different random seeds to observe variation caused by weight initialization. Fig. 2 shows the feature extraction architecture.

Phase 2: Evaluate the three CPN and three FSPN CL approaches with five different random seeds to observe variation caused by example ordering within each sequential

TABLE II
GLOBAL TASK ACCURACY (%) AFTER CL

	Epic 1	Epic 2	Epic 3
Static	53.2 ± 2.9	57.9 ± 1.9	61.2 ± 4.6
Fine Tune	21.3 ± 3.1	21.1 ± 6.4	20.5 ± 3.3
EWC	25.1 ± 6.2	26.8 ± 7.2	26.4 ± 7.9
1-Shot	56.0 ± 8.2	61.0 ± 3.4	55.8 ± 4.5
5-Shot	64.9 ± 4.1	67.4 ± 1.1	63.2 ± 2.9
10-Shot	**72.3 ± 3.1**	**68.0 ± 1.1**	**66.5 ± 2.1**

task data set for each of the five corresponding pretrained networks (25 results per method).

During CL the Static method learns new classes by adding the mean feature of 100 novel fault examples to the set of prototypes used for nearest prototype classification. Fine Tune saves the first batch of novel task data, creates a new randomized trainable prototype, and trains end-to-end for 20 epochs on the single batch with an Adam optimizer and learning rate of 10^{-3}. The EWC implementation mirrors Fine Tune but with an online EWC regularization term using $\lambda = 10^7$ and $\gamma = 0.9$. The n-Shot approaches collect the first n examples of novel classes and add them to the support set. The experiments track global accuracy on all tasks in addition to accuracy on each individual task.

When repeated for the three epics, these two phases produce 450 CL experiments to run on an NVIDIA GeForce RTX 3080 GPU and Intel Core i9-10900 CPU @ 2.80 GHz. For Phase 1, CPN models pretrain for 30 epochs on Task 1 using Adam optimization with a 10^{-4} learning rate. FSPN approaches pretrain for 30 epochs on Task 1 using Adam optimization with a 10^{-3} learning rate with a multiplicative decay of 0.8 per epoch.

IV. RESULTS AND DISCUSSION

The results match expectations; FSPN approaches score higher overall accuracy than CPN for all epics when using at least five shots of novel classes. Fig. 3 illustrates the advantage of FSPN as the models learn the tasks sequentially. In Epics 1 and 2, 1-Shot FSPN outperforms CPN by approximately 3% (see Table II). Adding more shots reduces the variance, and 10-Shot FSPN increases the accuracy compared to Static on Epics 1, 2, and 3 by 19%, 10%, and 5.3%, respectively.

Fine Tune suffers lower accuracy than Static since the weight updates cause the model to forget earlier task information. Interestingly, while EWC does offer marginal increases in accuracy over Fine Tune and prevents the model from forgetting Task 1 (see Fig. 4), it still falls considerably short of the Static model's overall performance. As Fig. 4 illustrates, Static maintains a relatively high overall accuracy due to high accuracy on the original task instead of comparable accuracy across individual tasks.

Recent image processing research might appear to contradict these results and indicate that Static CL outperforms FSPN [16]. However, these outcomes are not mutually exclusive. Ultimately, CL seeks good representations that can discriminate previously unseen—but related—classes. The data set, architecture, and training algorithm must work together to find these discriminative features. Meta-learning (e.g., FSPN) is one solution to regularize the training process, teaching the network to extract useful features from limited source information. Alternatively, if the base task data set contains a large number of diverse classes, regularization via meta-learning may be unnecessary. The diversity of base classes ensures that the representation will be generalizable. Thus, meta-learning may be advantageous for low-diversity data sets while traditional training remains effective for those with large numbers of classes. CM applications usually have only a few distinct conditions (e.g., eight total across all five tasks in this study), so meta-learning is appropriate and supported by the results in Table II. In contrast, few-shot image data sets may have 60 or more base classes, providing enough variation to learn generalizable features without meta-learning [16], although this remains an active area of research [19].

V. CONCLUSION

While classification of never-before-seen faults is vital for practical CM algorithms, state-of-the-art CM research focuses on (potentially offline) recognition of unknown conditions rather than online integration of novel faults into future predictions. Techniques like EWC could prevent models from forgetting earlier tasks when fine-tuning on new tasks, but extensive fine-tuning may not be possible on deployed CM hardware. Meta-learning approaches like FSPN can better guide training on the base task to find more generalizable representations. Experiments on a motor health condition data set demonstrate that FSPN approaches can consistently outperform the overall task accuracy of static, fine-tuning, and EWC methods, and this advantage might be accentuated by the limited class diversity in the base task. Future work should extend the FSPN architecture to include automatic detection of task boundaries (i.e., open set recognition) for unsupervised CL and quantitatively analyze varying numbers of classes in the base task.

REFERENCES

[1] S. Zhang, M. Wang, W. Li, J. Luo, and Z. Lin, "Deep learning with emerging new labels for fault diagnosis," *IEEE Access*, vol. 7, pp. 6279–6287, 2019.

[2] X. Yu, Z. Zhao, X. Zhang, Q. Zhang, Y. Liu, C. Sun, and X. Chen, "Deep-learning-based open set fault diagnosis by Extreme Value Theory," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 1, pp. 185–196, 2022.

[3] A. Bendale and T. E. Boult, "Towards open set deep networks," *IEEE CVPR 2016*, pp. 1563–1572, 2016.

[4] K. Saito, S. Yamamoto, Y. Ushiku, and T. Harada, "Open set domain adaptation by backpropagation," *IEEE ECCV 2018*, pp. 1–16, 2018.

[5] J. Li, R. Huang, G. He, S. Wang, G. Li, and W. Li, "A deep adversarial transfer learning network for machinery emerging fault detection," *IEEE Sensors*, vol. 20, no. 15, pp. 8413–8422, 2020.

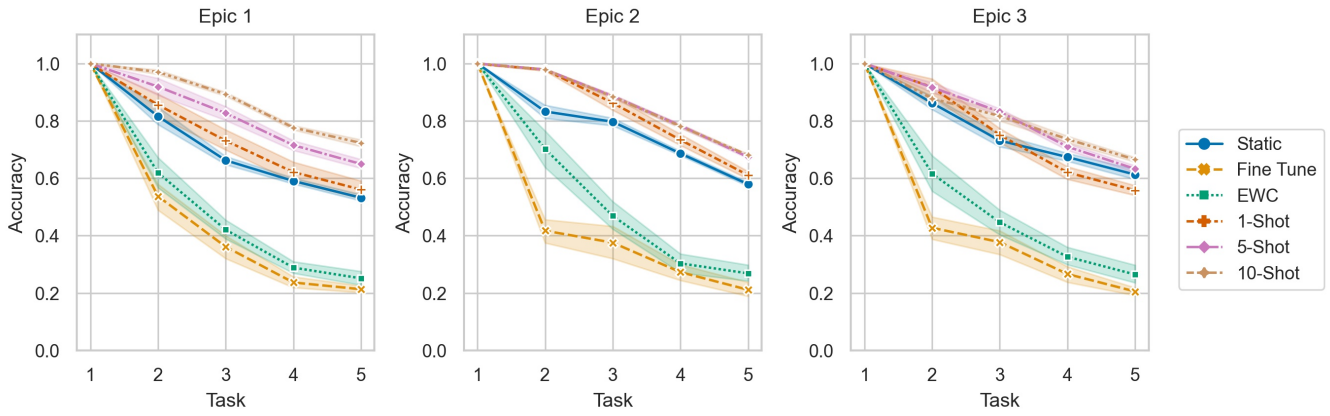

Fig. 3. Accuracy of each method on successive tasks within three distinct epics. FSPN with 5 and 10 shots consistently achieve higher sustained accuracy than CPN. In Epics 1 and 2, FSPN with only 1 shot exceeds the accuracy of all CPN approaches.

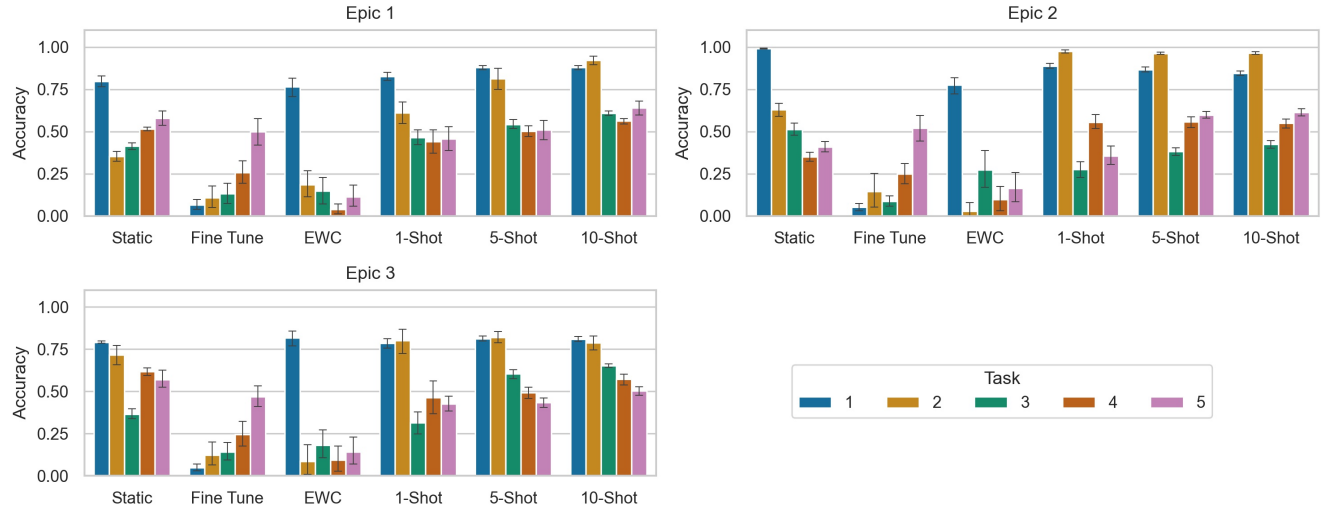

Fig. 4. Accuracy of each method on each task after all tasks have been learned. The accuracy of Fine Tune illustrates the forgetting of old tasks. EWC can preserve the original task information but inhibits future task learning. FSPN approaches, especially 10-Shot, experience a more even distribution of final task accuracy than CPN.

[6] J. Li, R. Huang, G. He, Y. Liao, Z. Wang, and W. Li, "A two-stage transfer adversarial network for intelligent fault diagnosis of rotating machinery with multiple new faults," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 3, pp. 1591–1601, 2021.

[7] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 2935–2947, 2018.

[8] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," *IEEE CVPR 2017*, pp. 5533–5542, 2017.

[9] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv:1503.02531*, pp. 1–9, 2015.

[10] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[11] H. Tercan, P. Deibert, and T. Meisen, "Continual learning of neural networks for quality prediction in production using memeory aware synapses and weight transfer," *Journal of Intelligent Manufacturing*, vol. 33, pp. 283–292, 2022.

[12] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory Aware Synapses: Learning what (not) to forget," in *Proceedings of the European COnference on Computer Vision (ECCV)*, 2018, pp. 139–154.

[13] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[14] H.-M. Yang, X.-Y. Zhang, F. Yin, Q. Yang, and C.-L. Liu, "Convolutional prototype network for open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, December 2020.

[15] S. Gidaris and N. Komodakis, "Dynamic few-shot visual learning without forgetting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[16] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, and Y. Gong, "Few-shot class-incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[17] H. Qi, M. Brown, and D. G. Lowe, "Low-shot learning with imprinted weights," *IEEE CVPR 2018*, pp. 5822–5830, 2018.

[18] J. Schwarz, J. Luketina, W. M. Czarnecki, A. Grabska-Barwinska, Y. W. Teh, R. Pascanu, and R. Hadsell, "Progress & compress: A scalable framework for continual learning," in *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, 2018.

[19] Y. Chen, Z. Liu, H. Xu, T. Darrell, and X. Wang, "Meta-Baseline: Exploring simple meta-learning for few-shot learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9062–9071.