Abstract

The foremost challenge in continual learning is devising strategies to alleviate catastrophic forgetting, thereby preserving a model’s memory of prior knowledge while learning new tasks. Knowledge distillation, a form of data regularization, is garnering increasing attention in the field of continual learning for its ability to constrain a model’s discriminative power for previous tasks by emulating the outputs of old task models while learning new tasks, thus mitigating forgetting. This paper offers a comprehensive survey of continual learning methods employing knowledge distillation within the realm of image classification. We inductively categorize these methods according to the source of knowledge utilized and provide a detailed analysis of the distillation solutions they employ. Furthermore, given the characteristic inability of continual learning to access historical data, we introduce a novel taxonomy for continual learning approaches from the perspective of auxiliary data usage. In addition, we have conducted extensive experiments on CIFAR-100, TinyImageNet, and ImageNet-100 across nine knowledge distillation-integrated continual learning methods, deeply analyzing the role of knowledge distillation in different continual learning scenarios to alleviate model forgetting. Our substantial experimental evidence demonstrates that knowledge distillation can indeed reduce forgetting across most scenarios.
Abstract—The foremost challenge in continual learning is devising strategies to alleviate catastrophic forgetting, thereby preserving a model’s memory of prior knowledge while learning new tasks. Knowledge distillation, a form of data regularization, is garnering increasing attention in the field of continual learning for its ability to constrain a model’s discriminative power for previous tasks by emulating the outputs of old task models while learning new tasks, thus mitigating forgetting. This paper offers a comprehensive survey of continual learning methods employing knowledge distillation within the realm of image classification. We inductively categorize these methods according to the source of knowledge utilized and provide a detailed analysis of the distillation solutions they employ. Furthermore, given the characteristic inability of continual learning to access historical data, we introduce a novel taxonomy for continual learning approaches from the perspective of auxiliary data usage. In addition, we have conducted extensive experiments on CIFAR-100, TinyImageNet, and ImageNet-100 across nine knowledge distillation-integrated continual learning methods, deeply analyzing the role of knowledge distillation in different continual learning scenarios to alleviate model forgetting. Our substantial experimental evidence demonstrates that knowledge distillation can indeed reduce forgetting across most scenarios.

Index Terms—Continual Learning, Incremental Learning, Knowledge Distillation, Catastrophic Forgetting

I. INTRODUCTION

Continual learning, an emergent field that addresses the need for adaptable intelligence, has garnered substantial interest recently. Standard deep learning methods learn a static data distribution from an established dataset, effectively targeting particular applications[1]. However, these methods encounter difficulties with data outside their trained distribution. Continual learning endeavors to enable models to assimilate new knowledge while preserving previous learning within dynamic data environments[2], [3]. This domain emphasizes the lifelong assimilation and refinement of knowledge across the lifespan of the model, analogous to human learning processes. In the literature, this discipline may also be called lifelong learning[4], [5], [6], incremental learning[7], [8], [9], or sequential learning[10], [11]. Continual learning’s significance stems from its capacity to facilitate the progressive acquisition of knowledge from a continuous influx of new data, thereby precluding the necessity for complete model retraining.

In contrast to traditional joint training, where the model has access to the entire dataset, continual learning models are often challenged by the phenomenon of catastrophic forgetting[10], [12] due to their inability to utilize historical data. This leads to a failure in retaining previously acquired knowledge upon the assimilation of new information. To tackle this issue, models must be designed to mitigate the loss of older knowledge while also facilitating the integration of new knowledge during the continual learning cycle. The capacity of a model to assimilate new knowledge and simultaneously preserve existing information is encapsulated by the stability-plasticity dilemma[13], [14]. The overarching aim of continual learning is to achieve an equilibrium between stability and plasticity—enabling the model to maintain critical knowledge and effectively incorporate emergent information.

A variety of strategies have been proposed to curb catastrophic forgetting within the realm of continual learning. Knowledge distillation (KD), serving as regularization-based method[8], [15], has progressively become a standard technique for alleviating the forgetting issue in continual learning by prompting the new task model to emulate the output of the old task model. Within the continual learning framework, the previously trained model acts as a “teacher” that progres-
sively relays its expertise to the “student” model which learns new tasks\[16], [17]. This process ensures the retention of learned information while facilitating the acquisition of novel tasks. KD for continual learning is often conceptualized as a form of self-distillation\[18], [19], wherein the teacher and student models share identical architectures except for their classification layers. The adoption of KD in conjunction with other methodologies, such as data replay, has seen a surge in popularity\[20], [15]. Figure 1 depicts the application of KD within continual learning framework.

Although KD has steadily become a prevalent method for countering catastrophic forgetting, its integration into continual learning practices, its efficacy in overcoming forgetting, and its comprehensive influence on the field require thorough exploration. Most contemporary surveys on continual learning primarily investigate the field from various methodological categorizations\[21], [8], [9], [15] and application domains\[22], [23], [24], [25], yet there is a notable scarcity of reviews analyzing the field through the lens of specific techniques aimed at mitigating the issue of forgetting in continual learning. In this survey, we undertake a scrutiny of continual learning methods that implement KD, primarily within the realm of image classification tasks. We have categorized continual learning methods that utilize KD based on the source of knowledge being distilled, primarily identifying three levels: logits-level, feature-level, and data-level. Concurrently, we conducted an in-depth analysis on how these methods apply distillation strategies to attenuate forgetting. Taking into account the constraints in continual learning scenarios where access to historical data is not possible, we have proposed a novel typology for the categorization of continual learning methods from the perspective of utilizing auxiliary data. To investigate the impact of KD on mitigating forgetting, we selected nine continual learning methods that incorporate KD and conducted extensive experiments on datasets such as CIFAR-100, Tiny-ImageNet, and ImageNet-100, thoroughly analyzing the role of KD in various continual learning scenarios. Overall, the principal contributions of this survey are delineated as follows:

- We present a comprehensive investigation of KD-integrated continual learning methods which mainly focus on image classification tasks. To our knowledge, this is the first systematic review in this area.
- We introduce a novel taxonomy to categorize the KD-integrated continual learning methods from two aspects: the nature of the distilled knowledge source and the approximation of old task data through auxiliary data. Our detailed analysis includes distillation loss formulations and delves into the prevalent techniques for combating forgetting.
- We conducted extensive experiments with nine KD-integrated continual learning methods on widely-adopted datasets across diverse continual learning scenarios to show the role of KD in alleviating forgetting.

This survey is structured as follows: Sec. II provides background on KD and continual learning. Sec. III elaborates on the problem formulation and established protocols for continual learning. Sec. IV categorizes KD-integrated continual learning methods by the distilled knowledge source and auxiliary data perspective. Sec. V details the experimental methodology and analyzes KD’s effects on continual learning extensively. Finally, Sec. VI outlines trends for future research in continual learning and Sec. VII concludes our survey.

II. PRELIMINARIES

In this section, we first explain the concept of knowledge distillation and then provide an overview of continual learning.

A. Knowledge Distillation

The fundamental aim of KD is to transfer knowledge\[26]. It follows a teacher-student schema\[16], wherein the student model is trained to emulate the outputs of the well-trained teacher model. The concept of KD was introduced by\[27], where knowledge transfer from teacher to student model is achieved by minimizing the Kullback–Leibler divergence\[28] between their outputs. Initially, the main application of KD is for model compression\[29], [30]. The growth in deep learning model size has increased computational and storage demands, limiting suitability for real-time processing and deployment on resource-constrained systems. KD addresses this by distilling large teacher models into more efficient student models, thereby easing computation and enabling deployment on lower-resource devices.

Except for model compression, the application of KD extends to various domains such as privileged learning\[31], [32], mutual learning\[33], [34], assistant teaching\[35], [36], self-learning\[37], [38], protective measures against adversarial attacks\[39], [40], and notably, continual learning\[41], [42]. In continual learning, KD transfers knowledge from old model while learning new tasks, aiding in the prevention of forgetting. Distinct from compression-centric distillation, in the context of continual learning, teacher and student models generally share identical architectures except the ultimate classification layer, giving rise to what is known as a self-distillation framework.

B. Continual Learning

Continual learning aims to learn from a continuous stream of data with changing distributions, retaining past knowledge even as new information is acquired. Unlike static train-and-deploy models, continual learning models are designed to grow and incorporate new skills and knowledge over time. However, these models frequently face catastrophic forgetting, where the introduction of new tasks without access to old task data causes loss of previously learned information.

Approaches to mitigating catastrophic forgetting in continual learning can be broadly classified into three main strategies. The first strategy is usually known as regularization-based methods\[43], [44], [45], which employs regularization techniques to steer model training toward a global optimum that accommodates all tasks. Another strategy is parameter isolation-based or architecture-based, where distinct parameters are allocated for each task during continual learning process. These approaches can be further divided into fixed\[46],
Fig. 2: Continual Learning Protocols. The number of new classes grows over time in CIL. Inference is performed without task IDs, whereas TIL requires task IDs. In DIL, the number of classes is fixed but their data distributions change. Balanced CIL is assumed where each class has equal training data. In Few-Shot CIL, abundant data is only available for the first task, with few samples per class in subsequent tasks. In Long-Tailed CIL, the data is imbalanced with differing amounts per class. The majority of continual learning settings are Offline CIL, allowing full passes over the current task data. Online CIL permits only a single pass over data as it arrives.
The continual learning model is conceptualized as without the opportunity to revisit data from preceding tasks. This technique is commonly referred to as replay-based methods [52], [53], [54].

### III. Continual Learning Setup

In this section, we first define the mathematical formulation of continual learning and introduce the relevant notation, followed by a description of various continual learning protocols.

#### A. Problem Formulation

In this study, we examine a continual learning paradigm wherein the model is exposed to a continuous data stream characterized by an evolving distribution. We define this stream as \( D_t = \{D_1, \ldots, D_i, \ldots, D_T\} \), where \( t \) denotes the index of a given task and \( T \) represents the total number of tasks. The tasks are introduced in a sequential manner, and the model is expected to learn from the current task \( D_t \) without the opportunity to revisit data from preceding tasks. The continual learning model is conceptualized as \( H = \{\theta, \phi\} \), which includes a feature extractor \( \theta \) and a classifier \( \phi \). For each task indexed by \( t \), the model encounters data \( D_t = \{X_t, Y_t\} \), wherein \( X_t \) signifies the set of inputs and \( Y_t \) corresponds to the associated labels, with each task comprising \( C_t \) classes. During the training phase on \( D_t \), inputs \( X_t \) are processed through \( \theta \) to feature vectors \( f = \theta(X_t) \). Subsequently, these features \( f \) are fed to \( \phi \) to generate the pre-softmax logits \( z = \phi(f) \). A fundamental challenge in this setting is the proclivity of \( H \) to catastrophically forget prior knowledge when past samples are not retained. To counteract this forgetting, certain strategies utilize a memory buffer, retaining a selection of samples from previous tasks to facilitate memory replay.

#### B. Protocols

This subsection offers an in-depth look at experimental protocols in continual learning, analyzing them from three perspectives. First, we categorize continual learning into Class Incremental Learning (CIL) [55], [56], [57], [58]. Task Incremental Learning (TIL) [7], [21] and Domain Incremental Learning (DIL) [59], [60], considering the updates to the model’s classification head and the need for task indicators during training and inference. Among these, CIL is currently one of the most prevalent protocols within the field of continual learning research. Based on CIL, we dichotomize continual learning into offline and online [61], [62], [63] CIL depending on the number of times a learning model accesses the data stream during training. Additionally, we classify the continual learning based on the count of training samples across classes, resulting in balanced, few-shot [64], [65], [66] and long-tailed CIL [67], [68]. The characteristics of the various continual learning protocols are illustrated in Figure 3.

**Class, Task and Domain Incremental Learning.** Class CIL is the most prevalent in continual learning, involving a series of tasks with unique labels to enable classification across all learned classes without task identifiers. The model’s classification heads grow with each new task. TIL resembles CIL but uses a task indicator during training and inference, allowing differentiation within the scope of a given task thereby simplifying class determination compared to CIL. DIL maintains a constant number of classification heads, facing the challenge of classifying samples from the same classes that exhibit varying distributions over time, delivered sequentially.

**Offline and Online CIL.** When not explicitly specifying “Online” CIL the term typically defaults to “Offline” CIL. Offline CIL allows models to iterate over the current task’s data stream until they reach convergence. This setup permits repeated training cycles for performance enhancement. In contrast, Online CIL restricts models to a one-time pass through incoming data, simulating real-world conditions where data is ephemeral and can only be preserved through physical capture. This single-pass restriction heightens the difficulty as models must quickly learn and integrate new information without revisiting the data.

**Balanced, Few-Shot and Long-Tailed CIL.** Existing methods in the field of continual learning often operate under the assumption that the data associated with incoming tasks is uniformly distributed, with each task providing an equivalent volume of training samples. We refer to this protocol as balanced CIL which is atypical in real-world settings. Few-Shot CIL (FSCIL) [64] blends few-shot learning with continual learning. In FSCIL, the model begins with extensive training data for the initial task, while subsequent tasks are introduced with only a few samples each, significantly fewer than the first task’s dataset. Long-Tailed CIL (LT-CIL) [67] acknowledges the complexity and imbalance of real-life data, addressing disparate data distributions across tasks. Tasks are sequenced by different sample volume, reflecting the long-tailed distribution found in real-world datasets, thus mirroring the diversity and imbalance typical in actual use cases.

### IV. Continual Learning with KD

In this section, we categorize KD-integrated continual learning methods from two perspectives. Initially, we classify these methods according to knowledge source, dividing them into three categories: logits-level, feature-level, and data-level. Furthermore, given that continual learning precludes access to data from previous tasks, rendering the model unaware of the old task data distribution, certain methods employ auxiliary data to approximate the perception of this distribution. Based on this characteristic, we divide continual learning methods into three paradigmatic types: exemplars-based, prototype-based, and data-free-based approaches. Figure 3 offers a detailed diagram of our classification framework.

#### A. Knowledge Source Perspective

KD facilitates the transfer of knowledge by having a student model emulate the knowledge of a teacher model. Knowledge in a model can be represented by its logits outputs or extracted features. Therefore, we can stratify the sources of knowledge into two principal tiers: logits-level and feature-level. In addition, generative models have been widely adopted in
Fig. 3: The schematic structure for continual learning with KD. Knowledge transfers from old tasks model (yellow network with feature extractor $\theta_{t-1}$ and classifier $\phi_{t-1}$) to new task model (red network with feature extractor $\theta_t$ and classifier $\phi_t$). Knowledge comes from different levels such as Logits-level, Feature-level and Data-level. To support old knowledge retention while acquiring new skills, auxiliary data about previous tasks is provided, which might include exemplars, prototypes, synthetic data, or nothing. We categorize KD-integrated continual learning methods based on the type of auxiliary data into three paradigms: Exemplars-based, Prototype-based and Data Free-based. Data-level distillation is common when the generative model also learns continually.

continual learning for their ability to create pseudo-historical data. The knowledge encoded by these generative models can be expressed through the data that they generate. Accordingly, KD-integrated continual learning methods can be grouped into three broad categories: logits-level, feature-level, and data-level. (as shown in Fig. 4).

1) Logits-level Distillation: Logits-level distillation primarily involves the student model assimilating knowledge by emulating the teacher model’s final output logits. These outputs generally constitute two types: normalized classification probabilities through a normalization function (e.g., softmax) and the raw, unnormalized logits. Consequently, we categorize logits-level KD methods in continual learning into two subcategories: Probability Matching and Logits Matching. Probability Matching is common, with the student aiming to match the teacher’s output probability distribution using loss functions like cross-entropy or KL divergence. In contrast, Logits Matching aims to synchronize the pre-softmax logit values, often using loss functions like L1 or L2 norms. Logits Matching imposes more stringent distillation constraints than Probability Matching. The right-hand section of Figure 5 provides a conceptual illustration, detailing the various loss functions employed in logits-level distillation.

Probability Matching. LwF [41] is the first to introduce KD in continual learning. It encourages the output probability distribution of new task learner model to approximate the output probability distribution of old task learner model in continual learning progress. It uses a modified cross-entropy loss to achieve the purpose:

$$L_{KD} = - \frac{C_{1:t-1}}{s} \sum_{i=1}^{C_{1:t-1}} \log(s(\pi(\hat{z}_i))) s(p_i) = \frac{p_i^{1/\tau}}{\sum_{j} p_j^{1/\tau}}$$

where $\hat{z}_i$ is output logit of old model, $z_i$ is output logit of new model, $\pi$ is softmax function, $C_{1:t-1}$ is total number of classes for old tasks, $s$ is a smooth function with temperature $\tau$. This form of KD was subsequently widely adopted in several works [69], [70], [71], [72], [61]. However, instead of the separate smooth loss function, these methods directly incorporate temperature in the softmax to perform distillation, as shown below:

$$L_{KD} = - \sum_{i=1}^{C_{1:t-1}} \pi(\hat{z}_i/\tau) \log\pi(z_i/\tau)$$

For simplicity and brevity, we hereafter denote the temperature-scaled softmax function as $\pi$ by default in all subsequent formulas.

iCaRL [42] is the first method that introduce rehearsal memory into continual learning and combine KD with memory replay to alleviate catastrophic forgetting. iCaRL proposes a
loss that inject classification into KD by interpreting network output logit as probability with a sigmoid function. It distills the old tasks knowledge with the post-sigmoid probability and classify new categories with a binary cross entropy classification loss:

$$L_{KD} = - \sum_{i=1}^{C_{l:t-1}} \sigma(\tilde{z}_i) \log \sigma(z_i) + (1 - \sigma(\tilde{z}_i)) \log(1 - \sigma(z_i))$$  \hspace{1cm} (3)$$

where \(\sigma\) is sigmoid function.

EELI [23] devises an end to end continual learning paradigm with a cross-distilled loss which incorporate cross entropy and distillation loss. It consolidates old task knowledge with individual probability matching for each old task which we call it task-wise distillation (TKD):

$$L_{KD} = - \sum_{i=1}^{C_t} \sum_{j=1}^{C_{l:t-1}} \pi(z_{ij}) \log \pi(z_{ij})$$  \hspace{1cm} (4)$$

where \(t\) is index of current task, \(C_t\) is number of classes for each task. SS-IL [74] proves the TKD is powerful when combined with Separated-Softmax classification layer.

D+R [75] combines distillation and retrospection to achieve a better balance between preservation on old knowledge and adaptation on new knowledge during continual learning. It distills knowledge from two teacher models, one is an intermediate expert model which is learned only from current new task data to adapt the target model to new task and meanwhile consolidates old task knowledge by KD on previous model.

$$L_{KD} = - \sum_{i=1}^{C_{l:t-1}} \pi(\tilde{z}_i) \log \pi(z_i) - \sum_{i=C_{l:t-1}+1}^{C_t} \pi(q_i) \log \pi(z_i)$$  \hspace{1cm} (5)$$

where \(C_{l:t}\) is number of classes until current task, and \(q_i\) is the response of expert model.

GD [76] designs a global distillation loss to maintain previous tasks knowledge by leveraging a large stream of unlabeled wild data which is easily obtainable and helpful to avoid overfitting to recent task. The global distillation loss comprises three parts: distillation from old tasks model \(M_{l-1}\), distillation from current task expert model \(M_{expert}\) and distillation of ensemble knowledge from \(M_{l-1}\) and \(M_{expert}\) to complete the missing knowledge with external wild data.

MBP [77] alleviates forgetting with a relaxed probability matching scheme that does not strictly match the post-softmax output distributions between the new and old task models. Instead, it matches the label priority vectors after sorting according to the probabilities vector from old and new task model. This helps to retains the semantic relevance between different classes learned by the model.

MC-OCL [78] proposes a Batch-Level Distillation loss to balances stability and plasticity when learning from incoming new task while preserving the performance on old tasks. It computes the probabilities of old task at mini-batch level in the warm-up training stage and prevent catastrophic forgetting by distill these mini-batch level probabilities in the joint learning stage. It performs KD in a extreme memory constraints setting without caching exemplars.

Logits Matching. DER++ [79] distills past experience by matching the network pre-softmax output logits sampled over the optimization trajectory rather than final optimized network’s output logits. It empirically proves that distillation on training trajectory can converge to a flatter loss landscape and achieve better model calibration. The knowledge is inherited with an Euclidean distance loss instead of the KL divergence to eliminate the information loss from squashing function like softmax.

$$L_{KD} = \|\hat{z} - z\|_2$$  \hspace{1cm} (6)$$

XDER [58] indicates there are two pitfalls in DER++. The stored logits trajectories in DER++ have a blind spot for future task cause they are learned from already observed data without any prognosis of future task data. XDER keeps the future part of pre-softmax logits update-to-date after learning new tasks with secondary information, and distills knowledge from the updated logits with the same loss function as used by DER++.

DMC [55] introduces a Deep Model Consolidation paradigm, which compacts knowledge from new task expert model and old task model with a double distillation loss. The compacted model is learned by distilling the concatenated knowledge with the help of auxiliary unlabeled data. The double distillation loss is defined as follows:

$$L_{KD} = \frac{1}{C_{l:t}} \sum_{i=1}^{C_{l:t}} (z_i - \hat{z}_i)^2 ,$$  \hspace{1cm} (7)$$

$$\hat{z}_i = \begin{cases} \tilde{z}_i & \text{if } 1 \leq j \leq C_{l:t-1} \\ \frac{1}{C_{l:t-1}} \sum_{j=1}^{C_{l:t-1}} \tilde{z}_j & \text{if } C_{l:t-1} < j \leq C_{l:t} \end{cases} ,$$  \hspace{1cm} (8)$$

where \(z_i\) represents the logits output of target compact model and \(\hat{z}_i\) is the concatenated logits from the old task model and the new task expert model. The term \(\tilde{z}_j\) refers to the logits output from the old task model if \(1 \leq j \leq C_{l:t-1}\), or from the new task expert model if \(C_{l:t-1} < j \leq C_{l:t}\).

R-DFCIL [80] introduces Hard Knowledge Distillation (HKD) and applies it to synthesized old-task data obtained through model inversion [51] to consolidate the distribution of old-task data. R-DFCIL found that using KL divergence with temperature-scaled softmax functions to align the outputs of
synthesized data on both the old and new models lacked sufficient constraint power. Hence, it applies an L1 loss constraint on the pre-softmax logits to enhance the consolidation of old knowledge.

\[ L_{KD} = \| \hat{z} - z \|_1 \]  

OCD-Net\[82\] devises online and adaptive pre-softmax logits distillation to counteract inherent response bias from the teacher model during offline KD, where the static teacher model can develop classification bias due to class imbalances in training data. OCD-Net updates the teacher model with the momentum update technique\[83\], ensuring the teacher model’s responses are up-to-date. Additionally, OCD-Net employs adaptive perception to modulate the distilled logits, enhancing the student model’s ability to learn high-quality responses from the teacher. The following is the online response distillation with adaptive perception:

\[ L_{KD} = \omega \| \hat{z} - z \|_2, \]

\[ \omega = \frac{\exp(\hat{z}_{gt} / \tau)}{\sum_{i=1}^{C_i} \exp(\hat{z}_i / \tau)}, \]  

where \( \omega \) is the quality score which is obtained by \[11\] where \( \hat{z}_{gt} \) represents the response corresponding to the ground truth class of the input sample.

2) Feature-level Distillation: Feature-level distillation seeks to impart knowledge derived from the internal representations produced during a network’s feature extraction phase. This type of methods can be divided into three subclasses based on the characteristics and location of the features within the network: Instance Feature Alignment, Hidden Feature Alignment, and Relation Alignment. Instance Feature Alignment targets the distillation of features from individual inputs presented to the classifier, which are usually converted into one-dimensional vectors. Hidden Feature Alignment focuses on the distillation of the intermediate layers’ features within the feature extractor, retaining spatial information inherent to the network’s structure. Relation Alignment, by contrast, aims to distill the intricate local or global relational dynamics among the features pertaining to multiple instances or prototype representations. The right half of Figure 5 presents a schematic representation that delineates the loss mechanisms inherent in diverse feature-level distillation.

**Instance Feature Alignment.** LUCIR\[84\] mitigates the adverse effects of catastrophic forgetting in continual learning with a distillation loss on normalized instance feature. It uses cosine distance to measure similarity between two normalized features and prevents features from changing and rotating dramatically with a constraint on feature cosine distance:

\[ L_{KD} = 1 - \langle \frac{\hat{f}}{\| \hat{f} \|}, \frac{f}{\| f \|} \rangle \]  

where \( \langle \hat{f}, f \rangle = \hat{f}^T f \) and \( f \) are features from old task model and current model. The feature cosine angle distillation is also adopted by \[85\], \[86\], \[87\].

GFR\[88\] restricts the change in generated features used for feature replay to prevent catastrophic forgetting. It trains a generator to memory features learned and replays them during continual learning to address imbalance problem on classifier. The feature extractor preserves old task knowledge with a feature distillation via L2 loss:

\[ L_{KD} = \| \hat{f} - f \|_2 \]  

\[89\], \[90\], \[91\], \[92\], \[93\], \[58\] follow it to use L2 feature distillation loss.

**Hidden Feature Alignment.** PODNet\[92\] fights catastrophic forgetting with a spatial-based distillation loss which applied on intermediate features and final instance feature. The PODNet\[92\] distills pooled intermediate feature after convolution layers and empirically finds that a spatial-based distillation pattern in height and width directions as Eq. [14] can achieve a better stability and plasticity trade-off. For the final output feature of feature extractor, it distills knowledge as Eq. [13]

\[ L_{KD} = \sum_{c=1}^{C} \sum_{h=1}^{H} \sum_{w=1}^{W} \| \hat{f}_{i,c,w,h} - \hat{f}_{i,c,w,h} \|_2 \]

\[ + \sum_{c=1}^{C} \sum_{h=1}^{H} \sum_{w=1}^{W} \| \hat{f}_{i,c,w,h} - f_{i,c,w,h} \|_2 \]

where \( c, h, w \) are index for intermediate feature in channel, height, width axes, \( i \) is index of intermediate layer.

TwF\[93\] employs a weighted feature distillation on model’s intermediate layers to address the issue of underutilized pretrained models in continual learning. Directly distilling these features through Euclidean distance may lead the model to excessively copy the pretrained model’s intermediate representations, compromising flexibility. Therefore, it uses an attention map as a binary mask to guide which aspects of the intermediate features to distill and which to exempt, as shown in Eq. [15]

\[ L_{KD} = \sum_{l=1}^{L} \| M(\hat{f}_l) \otimes (f_l - ReLU_m(\hat{f}_l)) \|_2 \]

where \( M(\cdot) \) is the module to calculate attention maps, \( \otimes \) is the Hadamard product, \( ReLU_m \) is the function of margin ReLU activation.

AFC\[96\] also adopts a weighted distillation solution for intermediate feature maps to confine changes of important features in continual learning. It estimates the importance of each feature map by minimizing upper bound of loss increases and gives a theoretical demonstration with Taylor approximation.

**Relation Alignment.** TPCIL\[97\] introduces a topology-preserving loss to maintain the manifold’s structure in feature space by penalizing alterations in class prototype relationships. It models the topological structure of the feature space using an Elastic Hebbian Graph (EHG) with nodes representing class prototypes and edges reflecting cosine distances between them. To retain prior knowledge, the newly formed EHG’s topology is constrained to resemble previous one.

ERL\[63\] uses an exemplar relation graph (ERG) to encode and preserve local relationships among cached exemplars, applying an exemplar relation distillation loss to maintain old knowledge. In ERG, each vertex represents an exemplar’s
Fig. 5: Different forms of KD loss at logits-level and feature-level. **Left:** Logits-level distillation losses. **Right:** Feature-level distillation losses. The directed solid line is the knowledge transfer direction in continual learning.
feature in feature space, with edges being unit vectors connecting the vertices. The model defines relationships between feature space exemplars through angles formed by triplets of vertices within the ERG. By constraining angle alterations during continual learning, ERL regulates feature space changes, reinforcing the memory of previously learned information.

Co2L[98] employs self-distillation on instance-wise relations to sustain learned representations. It incorporates an asymmetric supervised contrastive loss for representation learning, aiming to keep relations intact within batch samples while learning new task. The method computes the similarity between each sample and other samples in the batch, creating a similarity vector. This similarity relationship between the new and old model serves to distill knowledge from previous tasks. As this process is grounded in learned features, it is considered a feature-level distillation method.

MBP[77] focuses on preserving the model’s feature extraction behavior by maintaining the relative ordering of distances between instances in the feature space with an instance neighborhood-preserving loss. MBP calculates pairwise distances among all instances, identifying the top K nearest neighbors for each. By ensuring the order of proximal points remains consistent, MBP upholds semantic instance relationships while permitting development of more adaptable features, enhancing the model’s plasticity.

OCD-Net[82] integrates internal instance structure information from the teacher model using supervised contrastive learning[99]. It employs a contrastive relation distillation loss to cluster embeddings of the same class and separate those of different classes, utilizing cosine distance with temperature to cluster embeddings of the same class and separate those of different classes, utilizing cosine distance with temperature scaling. The teacher’s embedding serves as an anchor, guiding the student’s embedding toward the same-class embeddings found in the teacher model.

R-DFCIL[80] employs an angle-wise relational KD loss to learn representations for new classes while preserving representations for old classes. It achieves this by stabilizing the mutual spatial relationships of a triplet of new data within the feature space, which is transformed using a learnable linear layer. The relation KD to a triplet \((x_a, x_b, x_c)\) is as follows:

\[
L_{KD} = \| \cos \angle t_a t_b t_c - \angle s_a s_b s_c \|_1, \tag{16}
\]

\[
\cos \angle r_a r_b r_c = \langle e^{ab}, e^{cb} \rangle = \frac{r_a - r_j}{\| r_i - r_j \|_2} \tag{17}
\]

where \(t\) is transformed representations from old model, \(s\) is transformed representations from new model.

3) Data-level Distillation: Data-level distillation can be split into two types: Explicit Data Alignment and Implicit Data Alignment. Explicit Data Alignment entails distilling synthetic data produced by generative models. Contrastingly, Implicit Data Alignment focuses on distilling underlying information within the data, like attention maps or latent codes from generative models.

Explicit Data Alignment. MeRGAN[100] ensures memory retention in GANs during continual learning by aligning replay data generated by the model. Specifically, when given the same category and latent information as input to the generative model during learning new tasks, the content of the data generated by the new generative model should be consistent with that generated by the old generative model. This type of data alignment is a form of data distillation and has been adopted by other generative-model-based continual learning approaches, such as [101], [102].

Implicit Data Alignment. LwM[103] confronts forgetting by stabilizing the model’s attention. Using Grad-CAM[104], it calculates attention maps and preserves attention regions for old tasks while learning new ones, ensuring feature extracting capabilities are retained for prior knowledge. Additionally, LwM implements logits-level distillation similar to LwF, reinforcing old task knowledge consolidation. It follows Equation 18 to distill attention map where \(\hat{Q}\) and \(Q\) are vectorized attention map from old and new learning model.

\[
L_{KD} = \left\| \frac{\hat{Q}}{\|Q\|_2} - \frac{Q}{\|Q\|_2} \right\|_1 \tag{18}
\]

PGMA[101] alleviates forgetting by leveraging a VAE-based generator[105] to synthesize previous task data. The VAE features an encoder that compresses inputs into a latent representation and a decoder that reconstructs the inputs from this code. Since the generator itself must also continuously learn to synthesize new task data, PGMA applies distillation on the encoder’s sample-specific latent code during VAE’s continual learning. By regulating the latent space, this strategy helps curb forgetting within the VAE model.

B. Auxiliary Data Perspective

Continual learning methods frequently use auxiliary data to emulate past task distributions and address forgetting due to inaccessible old task data. This could entail storing exemplars, using prototypes to represent categories, or strategies that avoid direct old task data caching. This section categorizes these KD-integrated continual learning methods based on the auxiliary data they leverage into three paradigms: exemplars-based, prototype-based, and data-free-based approaches. Table lists methods under these paradigms, detailing their knowledge sources for KD and the specific distillation losses applied to mitigate forgetting.

1) Exemplars-based Paradigm: The exemplars-based paradigm combats forgetting by caching and replaying a subset of old task samples to mimic their original distribution. Pioneering methods like iCaRL and EEIL introduced this method, which has been widely adopted but provides only a partial remedy for forgetting. To further improve memory retention, it is essential to analyze inherent forgetting issues deeply and adopt custom solutions. We have identified common issues within continual learning and delineated strategies that are tailored to complement the exemplar-based paradigm in enhancing memory retention. These include optimizing KD, correcting classifier biases, improving feature representation learning, curating cache samples that match old task distributions more closely, and employing separate networks per task. It is worth noting that these techniques may also apply across different paradigms—topics. In the following, we will introduce these problems and techniques. Efficient KD. Exemplars-based approaches often concentrate on maximizing distillation’s potential to balance stability
<table>
<thead>
<tr>
<th>Paradigm</th>
<th>Methods</th>
<th>Distillation Level</th>
<th>Sub Distillation Type</th>
<th>KD Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rehearsal</td>
<td>iCaRL[42]</td>
<td>Logits</td>
<td>Probability Matching</td>
<td>Cross Entropy Loss</td>
</tr>
<tr>
<td></td>
<td>EEIL[73]</td>
<td>Logits, Feature</td>
<td>Instance Feature Alignment</td>
<td>Cosine Similarity Loss</td>
</tr>
<tr>
<td></td>
<td>BiC[69]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>D+R[75]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>KL Divergence Loss</td>
</tr>
<tr>
<td></td>
<td>FOSTER[107]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>ILOS[61]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>DualNet[108]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>POD-Spatial Loss</td>
</tr>
<tr>
<td></td>
<td>DyTox[109]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>RPS-Net[111]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>DCR[112]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>DER++[79]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>L2+R[114]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>MDMT-R[82]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>SSRE[83]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>PODNet[94]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>TwF[95]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>AFC[96]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>Co2L[98]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>ERL[65]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>MBP[77]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>SNCL[115]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>OCD-Net[82]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td>Prototype</td>
<td>Fusion[91]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>IL2A[89]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>PASS[90]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>FRoST[92]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>TPCIL[97]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>MEIL[85]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>LwF[41]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>BI-R[71]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>MC-OCL[78]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>DMC[55]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>CF-IL[116]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>GFR[88]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>ABD[117]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>NCDwF[118]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>R-DFCIL[80]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>MeRGAN[100]</td>
<td>Feature</td>
<td>Instance Feature Alignment</td>
<td>L2 Loss</td>
</tr>
<tr>
<td></td>
<td>LwM[103]</td>
<td>Data</td>
<td>Explicit data alignment</td>
<td>Cross Entropy Loss</td>
</tr>
<tr>
<td></td>
<td>PGMA[101]</td>
<td>Logs, Data</td>
<td>Explicit data alignment</td>
<td>L2 Loss</td>
</tr>
</tbody>
</table>

and plasticity in continual learning. They primarily seek to boost memorization by innovating in distillation techniques and content. EEIL[73] proposes a task-wise distillation loss. D+R[75] distills knowledge from both the old task model and an intermediate expert model for new task. GD[76] also trains an intermediate new task expert and uses global distillation with unlabeled data to distill from this expert, the old model, and their combined knowledge. ILOS[61] proposes a modified cross-distillation method to deal with online scenarios. PODNet[94] introduces a spatial-based feature distillation that maintains valuable features throughout continual learning. AFC[96] devises an importance-based feature distillation method that conserves critical features and allows more adaptability for less important ones during new task learning. Co2L[98] proposes an instance-wise relation distillation method that maintains memory by preserving local topological relationships between samples. GeoDL[87] projects the features onto low-dimensional manifold subspaces before distillation. COIL[57] proposes a bidirectional distillation loss which transfers knowledge forward and backward to
help model fast adapting and prevent catastrophic forgetting. MBP[74] aims to maintain model behavior with an instance neighborhood-preserving loss to prevent changes in instance relationships, alongside a label priority-preserving loss to avoid shifts in class priority. DER++[79] finds that distillation on logits trajectory in training process have a better effect on consolidating memory. XDER[56] improves on DER++[79] by updating the distilled logits based on the future task data’s influence on the logits, resulting in better performance. OCD-Net[82] uses online response distillation to counter teacher model bias, along with adaptive perception adjustments that enhance the teacher’s response quality.

**Classifier Bias Rectify.** Classifier bias occurs when models favor either new or old categories in continual learning. In exemplars-based paradigms, replaying a limited cache of old data alongside abundant new data inherently leads to an imbalanced scenario. Due to this discrepancy, classifiers tend to skew toward newer categories. EEL[73] recognizes this imbalance issue and alleviated it by adding a balanced fine-tuning stage, which has also been adopted in[61], [76]. BiC[69] formally pinpoints the classifier bias problem and improved it by training class correction parameters on a balanced validation dataset. RDICL[106] fixes the classifier bias issue through a dynamic threshold moving algorithm. WA[70] proposes weight aligning to correct the bias problem which does not require extra correction parameters calculated from a validation dataset like BiC[69]. LUCIR[84] addresses the imbalanced classification problem by normalizing features and increasing inter-class separation. GD[76] mitigates the bias by simulating feeding same data multiple times with scaled gradients during balanced fine-tuning stage. LV[110] designs a dual classifier, with one focused on new task feature learning and the other integrating knowledge from all tasks in a balanced manner. SS-IL[74] proposes the method of separated softmax combined with task-wise distillation to address bias. XDER[56] mitigates bias by adding restrictions on the activation of old and future tasks based on separated softmax.

**Informative Feature Learning.** Currently, representation learning in continual learning for image classification tasks is based on Empirical Risk Minimization with a cross-entropy objective. Several works aim to boost memorization by strengthening the model’s feature extraction capabilities, learning more representative and discriminative visual representations. These methods often leverage supervised contrastive learning[56], [82], [113], self-supervised learning[90], [82] and transformers[109], [110] to improve feature extraction. In addition, Co2L[98] proposes an asymmetric supervised contrastive loss to improve feature representation. DualNet[108] decouples commonly used supervised representation learning into two parts: unsupervised contrastive representation learning and supervised representation learning, and fully integrates both features. IL2A[89] proposes class augmentation based on Mixup[119] for representation learning.

**Coreset Selection.** Exemplars-based methods rely on a selection of old task data to represent its distribution. Choosing the most representative samples to approximate old task data distribution is crucial in reducing forgetting. Thus, some methods focus on identifying the best replay data, a principle has also been widely adopted by methods with or without KD. Mnemonics[72] obtains representative exemplars by parameterizing the exemplars and optimizing them with bilevel optimizations. GCR[113] constructs the replay buffer by approximating the gradients of current model over the learnt data. RM[120] enhances the sample diversity in episodic memory by measuring the classification uncertainty of a sample and its augmented variants. GSS[63] maximizes the diversity of exemplars by maximizing the variance of gradient directions from replay exemplars. Coreset[121] constructs the exemplar set by treating the selection process as a cardinality-constrained bilevel optimization problem.

**Parameter Isolation.** Assigning isolated parameters for each task is a common strategy to prevent forgetting in continual learning, which allocates distinct parameter subsets to different tasks, expanding model capacity and preserving task-specific knowledge. RPS-Net[111] divides the network into subnetwork paths guided by neural architecture search principles[122], aiming to find an optimal path for each individual task. DyTox[109] alleviates forgetting through dynamic expansion of task-specific tokens based on the Transformer’s attention mechanism[123]. FOSTER[107] adds task-specific modules to fit residual errors and then remove redundant parameters at feature compression stage with a balanced distillation strategy.

2) Prototype-based Paradigm: Using class prototypes to represent a class’s data distribution is a recognized machine learning technique. Adapted for continual learning, prototypes help encapsulate and track knowledge progression, supporting memory preservation. Abstraction of class traits into prototypes allows for a compact, representative model of prior knowledge, simplifying the integration of new information while mitigating memory degradation over time.

PASS[90] defines class prototypes as the mean of feature space data and introduces Gaussian noise for augmentation during new class learning, preventing classification bias toward new data. Meanwhile, PASS leverages self-supervised learning to learn more general features and KD to retain old and new model responses on new tasks, aiding old task memory preservation.

IL2A[89], like PASS[90], represents old class distributions with mean-based prototypes but also includes distribution variance for feature space data augmentation to protect old class boundaries during new class learning. IL2A uses KD for preserving old class feature extraction and label mixing augmentation to generate auxiliary data for more informative features.

FROST[92] conserves old class memory when learning new classes by replaying generative features drawn from a Gaussian distribution, which is defined by stored feature prototypes and variances of old class data. It also applies instance feature distillation to minimize feature drift toward the new class and mitigate forgetting.

Fusion[91] observes feature distribution drift with new class introduction, causing changes to prototypes. To gauge and correct prototype drift, it employs a DNN to parameterize either a Gaussian or variational model before classifier train-
ing. Additionally, it utilizes instance feature distillation to limit feature drift during continual learning.

MEIL[85] combats forgetting by replaying cached old task features. With feature space drift after learning new tasks, it transfers these cached features to updated feature space via a feature adaptation module. Besides, MEIL uses both logits and feature distillation to maintain the old model’s responses to new data during learning new tasks.

TPCIL[97] captures knowledge as an Elastic Hebbian Graph (EHG) in feature space, with nodes representing class prototypes and edges for inter-class cosine distances. It uses a topology-preserving loss to penalize changes to the EHG’s structure during incremental learning. For TPCIL, a base class training stage is essential for constructing a representative EHG.

3) Data Free-based Paradigm: Data-free paradigm methods avoid the explicit retention of data pertaining to old tasks. Instead, these methods use auxiliary data, such as unlabeled or synthetic samples that approximate old tasks, where synthetic samples can be created through model inversion or generative models. By training the updated model to match the previous model’s output on this auxiliary data while learning new tasks, old knowledge is preserved without keeping explicit historical data—providing a way to maintain model competence over successive learning stages.

LwF[41], the first data-free-based paradigm method, takes nothing related to old tasks but only distilling the response of new task data on the old model, thus preserving the memory of the old task. It is also the first method demonstrated that KD can alleviate forgetting.

Similarly, LwMi[103] suppresses forgetting by maintaining the attention regions of the old task model only with new task data and combining it with basic logits-level distillation.

DMC[55] utilizes publicly available unlabeled datasets and a double distillation loss that consolidates knowledge by distilling combined logits from new task expert models and old task models, without replaying any old task data.

ABD[117] synthesizes old task data via model inversion, capturing the feature distribution of old tasks. It uses an importance-based KD method to remember past tasks while maintaining model stability and plasticity.

R-DFCIL[80] also synthesizes old task data through model inversion and distills the old task’s feature extraction ability with synthetic data. It devises a relation-guided representation learning method with three losses: hard distillation on synthetic old data, feature relation distillation on new data, and local classification on new data to retain old features and adapt to new ones. After completing feature learning, it employs a classifier fine-tuning stage to correct bias with a global category-balanced classification loss.

MC-OCL[78] learns new tasks online without any replay data under extremely limited storage conditions based on Batch-Level Distillation. It involves a warm-up stage where the old model’s responses to new task data are cached, followed by joint training using these cached responses for distillation to preserve old task memory.

MeRGAN[100] combats forgetting with a conditional GAN-generated data replay. The generative model captures previous task knowledge, enabling synthesis of data resembling previous tasks’ distributions, which assists classifiers in continual learning. As the generator acquires new task knowledge, it must also retain previous information. MeRGAN addresses GAN forgetting with a data-level distillation strategy that aligns the generated data to preserve learned tasks.

PGMA[101] also adopts a generative model to synthesize replay data that helps prevent forgetting. It separates the continual learning model into a shared feature extractor and dynamic sample-specific parameters, learning a Dynamic Parameter Generator (DPG) alongside the feature extractor, rather than a unified model. To ensure the generator accumulates knowledge without forgetting, PGMA also aligns generated data and latent encodings via distillation.

V. Experiments

In this section, we experimented with nine KD-integrated continual learning methods on three standard image classification datasets to explore the role of KD in continual learning and assess its impact on mitigating forgetting in diverse scenarios.

A. Datasets

We select three image classification datasets widely used in continual learning field: CIFAR-100, TinyImageNet, and ImageNet-100, encompassing a range of image resolutions from 32×32, 64×64, to 224×224 pixels respectively. CIFAR-100[124], drawn from the “80 Million Tiny Images” collection[125], comprises 100 different categories, each with 500 training images and 100 testing images at a resolution of 32×32 pixels. ImageNet[125] is a large-scale dataset with 1.28 million training images and 50,000 validation images, spread across 1,000 categories. The ImageNet dataset has spawned various derivatives, which are sometimes inconsistently named in continual learning studies. We provide a unified nomenclature for clarity: ImageNet-1000, or ImageNet-Full, refers to the complete dataset, whereas ImageNet-100, introduced by iCaRL[42] and also known as ImageNet-Subset, features a fixed random seed shuffle of the first 100 ImageNet categories. Studies[96], [110], [109], [72], [94] commonly consider ImageNet-100 as a standard benchmark following the iCaRL setting. TinyImageNet[127], a downsized derivative of ImageNet created by Stanford University, has 200 categories with images downsampled to 64×64 pixels. Conversely, mini-ImageNet, presented in [128] for few-shot learning, consists of 100 categories at a resolution of 84×84 pixels.

B. Protocols and Scenarios

Our experiments focus on the Class Incremental Learning (CIL), using an offline and balanced protocol to evaluate different baseline methods. In balanced and offline CIL setup, two main strategies simulate data increment scenarios: the first divides the dataset into tasks with an equal number of classes for sequential learning. The second starts with preliminary base training on a subset of classes, followed by increments using remaining classes. To clearly describe these
scenarios, we adopt the notation from [8]. Specifically, the first scenario is notated as (A/B), where ‘A’ indicates the number of tasks and ‘B’ signifies the number of classes per task. For example, splitting CIFAR-10 into 5 sequential tasks with 2 classes each is notated as (5/2). The second scenario, involving base training followed by class increments, is expressed as (A/C-B). Here, ‘A’ stands for the total number of tasks, ‘C’ represents the initial base training class count (the first task), and ‘B’ indicates the number of classes for each subsequent incremental task. For instance, in the CIFAR-100 dataset, (11/50-5) denotes partitioning into 11 tasks—with initial base training on 50 classes, followed by an even distribution of the remaining 50 classes into 10 incremental tasks, each with 5 classes.

C. Metrics

There are many evaluation metrics used in continual learning for image classification tasks. iCaRL [42] introduces Average Incremental Accuracy (AIA), which captures the average of aggregate average accuracy after completing the learning process across all tasks. Meanwhile, GEM[129] introduces Average Accuracy (AA), Backward Transfer (BWT) and Forward Transfer (FTT) to assess the extent of catastrophic forgetting and the transferability of knowledge. Furthermore, RWalk[130] proposes Forgetting Measure (FM) and Intransi- gence Measure (IM) to evaluate the model’s average forgetting and inability to learn new tasks.

We perform an statistical analysis of the papers listed in Table I and found that Average Accuracy is the most prevalently utilized metric for Class-Incremental Learning (CIL). Other metrics, which are originally designed for Task-Incremental Learning (TIL), demonstrate limited applicability to CIL[8]. Consequently, we adopts Average Accuracy as the main metric. Average Accuracy evaluates average performance after learning t-th task which can be defined as

\[ AA_t = \frac{1}{t} \sum_{i=1}^{t} A_i \]

where \( A_i \) is the accuracy of task i evaluated on the test set after learning task t and \( i \leq t \). \( C_i \) denotes the number of classes contained in the task i, while \( C_{1:t} \) represents the cumulative sum of classes encompassed by all tasks learned up to the task t.

D. Baselines

We have chosen 9 KD-integrated continual learning methods from Table I to facilitate an exhaustive comparative analysis. These selected methods are LwF[41], LwM[103], IL2A[89], PASS[90], iCaRL[42], EEIL[73], BiC[69], LUCIR[84], and SS-IL[74]. Each method encapsulates distinct facets of the KD-integrated continual learning discussed in Section 3. More specifically, when categorized by the type of knowledge, LwF, iCaRL, EEIL, BiC and SS-IL are grouped as Logits-Level distillation techniques. Conversely, LUCIR, PASS, and IL2A are categorized under feature-level distillation, while LwM is considered a data-level distillation method. From an auxiliary data perspective, methods such as iCaRL, EEIL, BiC, LUCIR and SS-IL are part of the exemplars-based paradigm, whereas PASS and IL2A align with the prototype-based paradigm. Lastly, LwF and LwM subscribe to the data free-based paradigm. The selection of these methods is intended to ensure a broad-based evaluation across diverse distillation modalities and methodologies.

E. Training Details

We adopt ResNet-18[131] as the standard backbone network across all baseline methods due to its widespread use in image classification and continual learning benchmarks. ResNet-18, tailored for large-scale and high-resolution image datasets such as ImageNet, is modified to better suit smaller resolution datasets in our study. We adjust the initial convolutional layer while the original architecture remains for ImageNet-100 to accommodate its high-resolution images.

We implement all the methods selected in Section V-D using the FACIL framework, which is implemented in PyTorch. For all methods except IL2A and PASS, we use an SGD optimizer with an initial learning rate of 0.1, momentum of 0.9, and weight decay of 0.0002. IL2A and PASS employ an Adam optimizer with an initial learning rate of 0.001 and a momentum of 0.9. The learning rate is decreased every 45 epochs by a factor of 0.1 for IL2A and PASS. We use the herding sampling strategy for all exemplars-based methods. All experimental results are obtained by conducting five trials and calculating the mean and standard deviation of the results.
TABLE II: Comparison on Different Datasets. A growing memory with 20 exemplars each class is adopted for replay-based methods.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>CIFAR-100</th>
<th>TinyImageNet</th>
<th>ImageNet-100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario</td>
<td>(10/10)</td>
<td>(11/50-5)</td>
<td>(10/10)</td>
</tr>
<tr>
<td>LwF</td>
<td>24.99±1.12</td>
<td>11.95±0.93</td>
<td>18.17±2.40</td>
</tr>
<tr>
<td>LwM</td>
<td>26.10±1.48</td>
<td>24.99±1.12</td>
<td>18.60±1.19</td>
</tr>
<tr>
<td>IL2A</td>
<td>33.50±2.59</td>
<td>9.06±0.23</td>
<td>51.39±1.27</td>
</tr>
<tr>
<td>PASS</td>
<td>35.10±1.48</td>
<td>8.92±0.33</td>
<td>53.45±0.86</td>
</tr>
<tr>
<td>iCaRL</td>
<td>44.53±2.36</td>
<td>48.08±1.24</td>
<td>53.74±0.90</td>
</tr>
<tr>
<td>EEIL</td>
<td>39.92±1.76</td>
<td>39.54±1.70</td>
<td>42.70±0.82</td>
</tr>
<tr>
<td>BiC</td>
<td>48.09±2.70</td>
<td>45.12±2.19</td>
<td>55.21±0.82</td>
</tr>
<tr>
<td>LUCIR</td>
<td>44.50±1.12</td>
<td>44.87±0.70</td>
<td>47.35±1.78</td>
</tr>
<tr>
<td>SS-IL</td>
<td>44.57±1.80</td>
<td>41.02±0.53</td>
<td>49.80±0.43</td>
</tr>
</tbody>
</table>

TABLE III: Comparison on KD effect for 10/10, 11/50-5 incremental settings, exemplars-based methods use a growing memory with 20 exemplars each class for all scenarios.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>(10/10)</th>
<th>(11/50-5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>with KD</td>
<td>w/o KD</td>
<td>with KD</td>
</tr>
<tr>
<td>LwF</td>
<td>24.99±1.12</td>
<td>11.95±0.93</td>
</tr>
<tr>
<td>LwM</td>
<td>26.10±1.48</td>
<td>24.99±1.12</td>
</tr>
<tr>
<td>IL2A</td>
<td>33.50±2.59</td>
<td>9.06±0.23</td>
</tr>
<tr>
<td>PASS</td>
<td>35.10±1.48</td>
<td>8.92±0.33</td>
</tr>
<tr>
<td>iCaRL</td>
<td>44.53±2.36</td>
<td>48.08±1.24</td>
</tr>
<tr>
<td>EEIL</td>
<td>39.92±1.76</td>
<td>39.54±1.70</td>
</tr>
<tr>
<td>BiC</td>
<td>48.09±2.70</td>
<td>45.12±2.19</td>
</tr>
<tr>
<td>LUCIR</td>
<td>44.50±1.12</td>
<td>44.87±0.70</td>
</tr>
<tr>
<td>SS-IL</td>
<td>44.57±1.80</td>
<td>41.02±0.53</td>
</tr>
</tbody>
</table>

F. Results

In this section, we begin by conducting experimental comparisons of baseline methods within different scenarios and datasets. Subsequently, we delve into detailed experiments and in-depth analysis of the role that KD plays in mitigating forgetting.

1) On Scenarios: This section compares results on CIFAR-100 for four incremental learning scenarios: a (10/10) scenario without base training and three with base training covering half the categories—(6/50-10), (11/50-5), and (26/50-2). For methods using exemplars, we apply a dynamic buffer strategy that holds 20 exemplars per class. Figure 6 illustrates the Average Accuracy for each scenario.

Figure 6 reveals a notable performance difference across methods in different paradigm. A considerable performance gap is observed between data-free methods and exemplars-based methods in all scenarios, which can be attributed to whether exemplars have been cached or not. Intriguingly, the prototype-based strategy, which retains only class prototypes, demonstrates comparable or even superior efficacy to exemplars-based methods, especially in scenarios that involve in-depth analysis of the role that KD plays in mitigating forgetting. Moreover, comparative analysis of the (6/50-10), (11/50-5), and (26/50-2) scenarios reveals a general trend: more classes each enhance inter-class distinction, fostering that broader task scales imply fewer, larger tasks with task granularity leads to a decline in performance, suggesting incremental tasks correlate with lower performance. Finer

2) On Datasets: In this section, we assess baseline methods on datasets with varying resolutions: CIFAR-100 (100 classes, 32×32), TinyImageNet (200 classes, 64×64), and ImageNet-100 (100 classes, 224×224). We compare performances in two scenarios: a 10-task scenario without base training and an 11-task scenario with base training. For exemplar-based methods, we employ a growing memory that holds 20 exemplars per task.

Table II presents the empirical outcomes of several baseline strategies across datasets with varying resolutions: CIFAR-100, TinyImageNet, and ImageNet-100. In the 10-task trials without base training, the BiC approach outperforms its counterparts on all datasets. For the 11-task series with base training, iCaRL stands out as the top performer on both CIFAR-100 and TinyImageNet, while on ImageNet-100, the prototype-based PASS strategy exceeds the exemplars-based iCaRL technique, achieving the highest results.

Across the datasets, methods such as LwF and LwM, which rely heavily on KD, show superior performance without base training compared to scenarios where it is employed. Conversely, other methods generally achieve better results with the inclusion of base training, except for BiC and SS-IL on ImageNet-100, where they perform without it. Notably, the relative benefit of base training for BiC and SS-IL diminishes as the resolution of the dataset increases.

3) On KD Effect: This section delves into KD’s role in continual learning methods by conducting ablation studies that omit the KD loss. We analyze nine approaches, predominantly using a single distillation loss function, except for LwM, which employs two forms of distillation loss. We remove the logits-level distillation loss used by LwF, iCaRL, and BiC to assess its impact. For EEIL, we eliminate its task-wise KD loss employed in both balanced and unbalanced training phases. SS-IL is similarly adjusted by removing its task-wise KD loss that complements separated softmax. For IL2A and PASS, we abrogate the L2-based feature distillation loss crucial for instance feature alignment. LUCIR’s modification entails removing the cosine feature distillation loss. In LwM, we exclude only the attention map distillation loss—which is considered as implicit data distillation—to isolate the effect of the attention distillation, keeping its logits-level distillation intact.

Table III highlights how KD impacts different continual...
learning approaches on CIFAR-100 in (10/10) and (11/50-5) scenarios. Our observations reveal that KD can alleviate forgetting except for a few special cases. In the (10/10) scenario, iCaRL shows a marked performance increase after removing KD, and LUCIR experiences a slight improvement. However, without KD, other methods exhibit varying degrees of performance drops. In the (11/50-5) setup, the result is more consistent—every method suffers from performance deterioration without KD components. LwF’s reliance on KD is apparent, removing its distillation loss equates to basic fine-tuning on new tasks and leads to a stark performance drop. LwM incurs a obvious decline in the scenario without base training and a slight drop in the scenario with base training, indicating the specific benefit of attention map distillation under scenarios without base training. IL2A and PASS, heavily dependent on their distillation strategies, experience a significant performance drops when their distillation losses are removed. Similarly, BiC and SS-IL also show a strong reliance on distillation, with evident declines in both scenarios once distillation is deactivated. EEIL and LUCIR show a modicum of resilience to distillation removal, with only minor performance changes. Interestingly, LUCIR slightly improves without distillation in the (10/10) scenario, but performs less well in the (11/50-5) setup. This suggests that cosine similarity-based feature distillation is more critical when base training is incorporated. iCaRL stands out among the evaluated methods, showing notable improvement without KD in the (10/10) scenario. Yet, this is reversed in the (11/50-5) scenario, where its performance drops without distillation. These varied results highlight KD’s complex role and its diverse impact across different continual learning methods and scenarios.

The whole results indicate that prototype-based and data free-based approaches, which do not utilize exemplar replay, rely heavily on KD to prevent forgetting. Removing distillation loss typically leads to pronounced deficits in knowledge retention for these methods. Conversely, for exemplars-based methods, while KD can serve to mitigate forgetting, its impact is neither consistent nor universally positive. Notably, under the (10/10) scenario, KD integration could even hamper a model’s ability to remember past tasks.

4) On Base Training for KD: Our previous experiments demonstrate KD’s benefits are more pronounced with base training. To investigate the influence of base training on the efficacy of KD, this section will employ varying numbers of classes for base training, examining three scenarios: (9/60-5), (7/70-5) and (5/80-5).

Table IV: Continual learning performance with and without KD when using different numbers of classes for base training.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>9/60-5</th>
<th>7/70-5</th>
<th>5/80-5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>with KD</td>
<td>w/o KD</td>
<td>with KD</td>
</tr>
<tr>
<td>LwF</td>
<td>25.35±5.41</td>
<td>8.21±1.49</td>
<td>34.59±2.92</td>
</tr>
<tr>
<td>LwM</td>
<td>24.19±4.52</td>
<td>25.35±5.41</td>
<td>32.76±3.34</td>
</tr>
<tr>
<td>IL2A</td>
<td>54.86±1.09</td>
<td>4.64±0.12</td>
<td>58.13±1.05</td>
</tr>
<tr>
<td>PASS</td>
<td>56.67±0.24</td>
<td>4.59±0.14</td>
<td>59.25±0.45</td>
</tr>
<tr>
<td>iCaRL</td>
<td>56.45±0.83</td>
<td>51.30±0.33</td>
<td>60.51±0.47</td>
</tr>
<tr>
<td>EEIL</td>
<td>45.13±0.78</td>
<td>41.99±1.28</td>
<td>47.91±1.82</td>
</tr>
<tr>
<td>BiC</td>
<td>58.00±2.23</td>
<td>50.00±1.64</td>
<td>61.15±1.02</td>
</tr>
<tr>
<td>LUCIR</td>
<td>48.75±1.35</td>
<td>48.36±1.09</td>
<td>53.76±1.19</td>
</tr>
<tr>
<td>SS-IL</td>
<td>51.12±0.46</td>
<td>43.24±0.50</td>
<td>52.44±0.26</td>
</tr>
</tbody>
</table>

Table V: Comparison on different KD losses under exemplars replay paradigm. CE is Cross Entropy loss, KL represents KL divergence loss, L2 is L2 distance loss, CS represents Cosine Similarity loss.

<table>
<thead>
<tr>
<th>Methods</th>
<th>(10/10)</th>
<th>(11/50-5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ CE</td>
<td>38.43±1.08</td>
<td>47.10±0.89</td>
</tr>
<tr>
<td>+ KL</td>
<td>33.99±0.90</td>
<td>(7.11)</td>
</tr>
<tr>
<td>+ L2</td>
<td>43.35±0.97</td>
<td>(6.75)</td>
</tr>
<tr>
<td>+ CS</td>
<td>41.39±1.69</td>
<td>(9.29)</td>
</tr>
</tbody>
</table>

Table IV details how KD impacts continual learning with varying base classes. It is observed that all methods, except LwM, demonstrate enhanced resistance to forgetting with KD across the three scenarios. Without its attention distillation loss, LwM’s performance is comparable to LwF’s, and omitting this loss led to improvements in all three scenarios, suggesting it may not effectively combat forgetting when the initial task model already possesses strong feature representation. Methods like LwF, IL2A, and PASS remain KD-dependent, whereas iCaRL, BiC, and SS-IL consistently counteract forgetting with KD. Furthermore, the effectiveness of LUCIR’s cosine distillation loss against forgetting increases as the number of classes involved in base training grows.

Overall, KD’s effectiveness in reducing forgetting is notably enhanced by base training. However, not all methods see increasing benefits with more base training classes. Base training endows the initial task model a stable feature representation, improving the original teacher model’s knowledge quality, which is crucial for KD-integrated continual learning methods.

5) On KD Losses: To explore the effects of different KD losses and further understand KD’s role in continual learning, we compare several KD losses alongside a basic replay paradigm using herding algorithm for a growing memory with 20 exemplars per class. We evaluate cross-entropy, KL divergence, L2 distance loss at the logits level, as well as L2 distance-based and cosine similarity-based instance feature alignment losses. Through this comparison, we aim to gain deeper insight into each KD loss’s specific contribution to continual learning.

Table V shows that integrating KD with exemplars replay doesn’t consistently prevent forgetting and can even reduce performance. Specifically, distillation at the logits level has been observed to result consistently in a detrimental effect on performance. These findings align with observations from other research efforts, such as those cited in [132] and [8].
In scenarios without base training, the negative impact from logits-level distillation becomes more conspicuous, with the cross-entropy loss exhibiting a marginally lesser extent of damage and the KL divergence loss inflicting the greatest harm. At the feature level, L2 distance for instance alignment appears more effective in maintaining feature consistency, while cosine similarity yields only marginal benefits. When base training is employed, the adverse effects of logits-level distillation seem to recede, likely due to robust feature representations established by initial learning on a larger set of base classes. Under these conditions, the performance decrease is less pronounced. Interestingly, the more lenient constraint imposed by cosine similarity at the feature level seems to be advantageous, facilitating better retention of learned features.

VI. TRENDS OF CONTINUOUS LEARNING

This section analyzes future trends of continual learning across four aspects—data, scenarios, algorithms, and applications—and explores how KD might drive its progress.

A. Data

The vast diversity of real-world data presents challenges for continual learning algorithm development. While traditionally centered on images\cite{9},\cite{20}, recent years have seen expansion into text, speech, and video modalities. In the text domain, natural language processing has become an important direction, including continual learning for tasks like machine translation\cite{133},\cite{134}, text classification\cite{135},\cite{136}, dialogue systems\cite{137},\cite{138} and text generation\cite{139},\cite{140}. Some recent works have applied continual learning to audio classification\cite{141},\cite{142} and speech recognition\cite{143},\cite{144}. For video data, time-series based continual learning has become a hot topic in areas such as video object classification\cite{145},\cite{146} and detection\cite{147}, video super-resolution\cite{148}, etc. Furthermore, multi-modal continual learning has emerged as a new subfield, demanding both intra- and inter-modality knowledge consolidation and transfer, such as joint audio recognition and visual understanding\cite{149},\cite{150}, and learning from multiple sources like text and images\cite{151},\cite{152}. Building unified learning mechanisms to handle heterogeneous inputs across modalities remains one of the core challenges for future research.

B. Scenario

Real-world scenarios complexity and dynamism challenge the primarily idealized settings of most current continual learning research. Typically, tasks have equal numbers of categories and ample offline training data—unlike in actual applications, where task boundaries blur, category counts vary, and data distributions are imbalanced. Real-time streaming further complicates matters by limiting offline training. Therefore, investigating continual learning settings that better resemble reality is crucial. Some studies have started to explore few-shot\cite{64},\cite{65} or even zero-shot\cite{153},\cite{154} continual learning to reduce dependence on new data. Handling long-tailed\cite{67} data distributions is another new direction. Furthermore, task categories are often unknown in advance in many applications, giving rise to the Task-Free continual learning paradigm\cite{155},\cite{156}. Online continual learning\cite{61},\cite{62} has also become popular to simulate streaming data where training data can only be observed in one pass. Considering that many real-world data lack labels, unsupervised\cite{157},\cite{158} and semi-supervised\cite{159},\cite{160} continual learning also receive widespread attention. Real-world scenarios exhibit great complexity, dynamism, and open-endedness. Bridging the gap between controlled laboratory settings and real-world conditions is a core challenge for advancing continual learning research.

C. Algorithm

Unraveling catastrophic forgetting during continual learning, which is not yet fully understood, remains a primary focus. Recent works aim to demystify forgetting through theoretical analysis\cite{161},\cite{162}. While experience replay is a common countermeasure to forgetting, its computational cost has prompted research into exemplar-free solutions\cite{163},\cite{164},\cite{165}. Utilizing pretrained models for parameter-efficient fine-tuning is gaining attention due to their strong feature extraction capabilities\cite{166},\cite{167},\cite{159},\cite{168}. Generating pseudo-data of previous tasks using generative models like diffusion model\cite{169},\cite{170} still provides a promising direction. Beyond conventional forgetting-related research, combining continual learning with other learning paradigms represents an important trend. Federated learning, which focuses on decentralized learning to protect privacy, benefits from continual adaptation to the increasing local data\cite{171},\cite{172},\cite{173}. Continual reinforcement learning\cite{174},\cite{175},\cite{176} unifies the strength of reinforcement learning in adapting to dynamic environments and continual learning in handling non-stationary tasks. Moreover, the synergy between graph learning and continual learning sparks new investigations\cite{177},\cite{178},\cite{179}. The twin trends of delving into the nature of forgetting and crafting continual learning approaches within broader learning frameworks are expanding both theory and application prospects in the field.

D. Application

Continual learning initially focused on classification tasks, including image, text, and speech recognition. As the field matures, research has expanded to encompass other tasks, such as object detection\cite{180},\cite{181},\cite{182} and semantic segmentation\cite{183},\cite{184},\cite{185} for computer vision, and machine translation\cite{133},\cite{134}, dialogue systems\cite{137},\cite{138}, and text generation\cite{139},\cite{140} for natural language processing. Building on these foundational tasks, continual learning is seeping into various application domains. In autonomous driving and robotics, methods have been proposed to enable continual optimization of lidar odometry\cite{186} and SLAM\cite{187},\cite{188}. In medical imaging, continual learning has been applied to pathological image analysis\cite{189} and segmentation\cite{190}. Research is also burgeoning in applying continual learning to embodied AI\cite{191},\cite{192},\cite{193}, text recognition\cite{194},\cite{195}, visual question answering\cite{196},\cite{197},\cite{198}, and sign
language recognition is also emerging. The expansion from basic tasks to a wider spectrum of applications underscores continual learning’s increasing potential for real-world impact.

E. KD for Continual Learning

Experimental findings confirm KD’s prowess in mitigating forgetting during continual learning. The key to addressing forgetting with KD is pinpointing the historical knowledge most critical for retention. Methods like DER++[79] and XDER[50] have shown that well-chosen imitation targets can significantly boost task memory preservation. Additionally, online distillation methods like OCD-Net[82] reinforce this effect. Therefore, discerning high-quality knowledge for distillation stands as a crucial focus area to counteract forgetting in continual learning. Furthermore, the use of pretrained models in continual learning is gaining traction[152]. Tailoring KD to the specific characteristics of task to tackle forgetting is an expanding trend in continual learning research.

VII. CONCLUSION

Our study performs an in-depth investigation of continual learning methods utilizing KD. We introduce two innovative classification frameworks based on knowledge sources and auxiliary data utilization, presenting an overview of different distillation losses and strategies to prevent forgetting. Extensive experiments on nine KD-integrated methods across three datasets in varied scenarios highlight KD’s role in continual learning and its potential to mitigate forgetting. Additionally, we explore the future directions for continual learning across data types, scenarios, algorithmic approaches, and practical applications, and consider how KD might further contribute to these developments. This paper aims to deepen the understanding of KD’s impact in continual learning and to inform ongoing research in the domain.

REFERENCES
