Few-Shot Incremental Learning with Continually Evolved Classifiers

Chi Zhang[1]*, Nan Song[1]*, Guosheng Lin[1]†, Yun Zheng[2], Pan Pan[2], Yinghui Xu[2]
[1] Nanyang Technological University, Singapore [2] Alibaba DAMO Academy
{chi007,nan001}@e.ntu.edu.sg, gslin@ntu.edu.sg
{zhengyun.zy,panpan.pp}@alibaba-inc.com, renji.xyh@taobao.com

Abstract

Few-shot class-incremental learning (FSCIL) aims to design machine learning algorithms that can continually learn new concepts from a few data points, without forgetting knowledge of old classes. The difficulty lies in that limited data from new classes not only lead to significant overfitting issues but also exacerbate the notorious catastrophic forgetting problems. Moreover, as training data come in sequence in FSCIL, the learned classifier can only provide discriminative information in individual sessions, while FSCIL requires all classes to be involved for evaluation. In this paper, we address the FSCIL problem from two aspects. First, we adopt a simple but effective decoupled learning strategy of representations and classifiers that only the classifiers are updated in each incremental session, which avoids knowledge forgetting in the representations. By doing so, we demonstrate that a pre-trained backbone plus a non-parametric class mean classifier can beat state-of-the-art methods. Second, to make the classifiers learned on individual sessions applicable to all classes, we propose a Continually Evolved Classifier (CEC) that employs a graph model to propagate context information between classifiers for adaptation. To enable the learning of CEC, we design a pseudo incremental learning paradigm that episodically constructs a pseudo incremental learning task to optimize the graph parameters by sampling data from the base dataset. Experiments on three popular benchmark datasets, including CIFAR100, miniImageNet, and Caltech-USCD Birds-200-2011 (CUB200), show that our method significantly outperforms the baselines and sets new state-of-the-art results with remarkable advantages.

1. Introduction

Deep Convolutional Neural Networks have gained remarkable success in many computer vision tasks [10, 19, 22, 38, 55], stemming from the availability of big curated datasets, along with unprecedented computing power. However, a classification model that is trained by supervised learning can only make predictions on a set of pre-defined image categories. If we want to extend a trained model on new classes, a large amount of labeled data for new classes as well as data from old classes are both necessary for network finetuning, which inevitably hinders its real-world applications. If the dataset of old classes is no longer available, directly finetuning a deployed model with new classes can lead to the notorious catastrophic forgetting problem that knowledge about old classes is quickly forgotten [11, 17, 34]. In contrast to machine learning systems, humans are readily able to learn a new concept with few examples without forgetting old knowledge. The gap between humans and the machine learning algorithms fuels interest in few-shot class-incremental learning (FSCIL) [39], which aims to design machine learning algorithms that can be continually extended to new classes with only a few data points. The challenge of FSCIL lies in that the scarcity in the data of new classes will not only cause severe overfitting but also exacerbates the catastrophic forgetting problem of old classes. In this paper, we undertake the task of few shot incremental learning and consider to solve the aforementioned problems from two aspects.

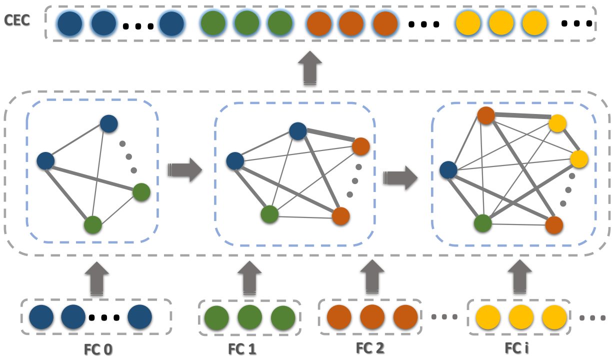


Figure 1: Illustration of our proposed continually evolved classifiers for FSCIL. We employ a graph model to adapt the classifier weights learned on individual sessions for the prediction over all classes.

[*] indicates equal contribution.
[†]Corresponding author: G. Lin (e-mail: gslin@ntu.edu.sg)

First, as the data from base classes and new classes are severely unbalanced, we propose to decouple the learning of representations and classifiers for the FSCIL problem. Specifically, the model only learns the representations in the first session where abundant data from base classes are available, and in the new sessions, we fix the network backbone and only adapt the classifier for new classes. Thus, we can avoid the overfitting problem as well as the catastrophic forgetting problem in the representations. By doing so, we demonstrate that a pre-trained network backbone based on data from base classes plus a class mean classifier can beat state-of-the-art approaches.

Second, as the classifiers are always learned from the classes in individual incremental sessions, they can only provide discriminative information for classifying internal categories, while incremental learning aims to learn models that can apply to all classes. As a result, even if a classifier can learn a well-separated decision boundary for the previous classes, it may lose the generalization ability when more novel classes are involved. For example, a vehicle-related representation *wheel* is chosen by the classifier as a discriminative representation to distinguish the categories *car*, *dog* and *cup* in the current classification task. However, when a new category *trunk* is involved in the new sessions, such representation may not be discriminative enough to classify all categories. Therefore, the incremental learning algorithm should have the flexibility to adjust the classifiers in previous sessions based on the overall task context to undertake the entire classification task. To this end, we present a Continually Evolved Classifier (CEC) that can progressively adapt the classifier weights based on current and history tasks. At the core of our network is a classifier adaptation module which uses a graph attention network (GAT) [41] to adapt the classifier weights learned on each task. By contextualizing individual classifier weights over the global task, the adapted classifiers highlight the discriminative representations in the backbone and generate better decision boundaries over all involved classes.

To enable the learning of the proposed continually evolved classifier, it is important to optimize the graph model under an incremental learning scenario. However, in incremental learning, datasets from different training sessions can never be accessed simultaneously for training. To overcome the issue, we propose a pseudo incremental learning paradigm, where we episodically construct a pseudo incremental learning task from the dataset in the base session to simulate the incremental learning scenario for training. Our design takes inspirations from the meta-learning paradigm [42]. In each pseudo incremental learning episode, we first sample a set of classes from the base dataset to play the role of the base classes, then we sample another group of classes to play the role of incremental classes to learn the model. However, as the pre-trained backbone has already learned feature representations that can well classify the base classes, directly using the sampled classes from the base dataset for learning may bypass the GAT and thus fail to impose context knowledge. We solve this problem by randomly rotating the sampled pseudo incremental classes with a large angle to synthesize new classes. In this way, we intentionally synthesize unfamiliar classes at training time to enforce context knowledge propagation in the graph model. Once the graph model is learned, we can use the graph model to update the classifier weights learned in incremental sessions.

To validate the effectiveness of our proposed method, we conduct comprehensive experiments on multiple benchmark datasets. The contribution of this work is summarized as follows:

- We adopt a decoupled training strategy for representation learning and classifier learning to avoid knowledge forgetting and overfitting in the backbone.
- We propose a continually evolved classifier that employs a graph model to combine classifiers learned on individual sessions for incremental learning.
- To enable the learning of the graph model in CEC, we design a pseudo incremental learning paradigm.
- Experiments on the CIFAR100, CUB200 and mini-Imagenet datasets show that our method significantly outperforms the baselines and sets new state-of-the-art performance with remarkable advantages.

2. Related Work

Few-Shot Learning. Few-shot learning aims to learn a model that can classify unseen images when only training from scarce labeled training examples [5,48]. Research literature on few-shot learning demonstrates great diversity. Optimization-based methods [8,16,24,28,29,36,37,50] and metric-based methods [9, 12, 35, 42, 46, 47, 51, 52] are two main lines of efforts. Optimization-based methods aims to design efficient learning paradigm that enables fast network adaptation given limited data [8, 16, 28, 29, 37]. Our work is more related to metric-based approaches, where a pre-trained backbone is used to encode data, and a distance metric, such as negative L2 distance [35], cosine similarity [42] and DeepEMD [51, 52], is used to measure data similarity and compute scores. Chen *et al*. [5] presents a baseline for few-shot classification that first pre-trains a backbone based on data from seen classes, and only finetunes the classifier for novel classes, which shares similarity with our decoupled training strategy. Apart from image classification, few-shot learning has also been applied to dense prediction tasks [6,21,23,53,54] and object detection [45].

Incremental Learning. Incremental learning (IL) is an active machine learning task that aims to learn new knowledge continually without forgetting [3, 4, 7, 25]. Recent

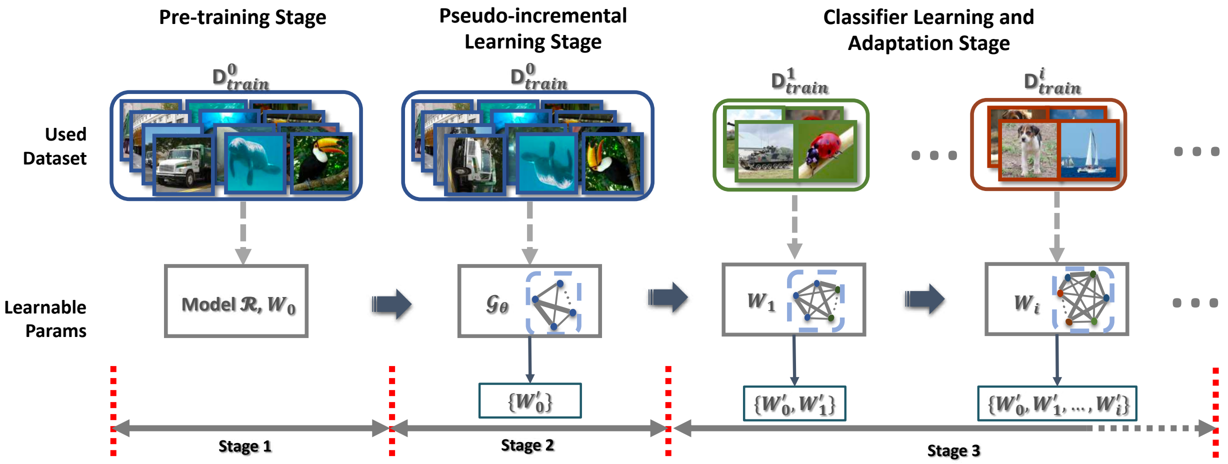


Figure 2: Our framework for few-shot incremental learning mainly includes three stages: (1) the feature pre-training stage to learn the backbone model $\mathcal{R}$ using the training data in the base session $\mathcal{D}^0_{train}$, (2) the pseudo incremental learning stage that trains the graph model $\mathcal{G}_\theta$ by sampling pseudo incremental tasks from $\mathcal{D}^0_{train}$, and (3) the classifier learning and adaptation stage using few-shot training data $\mathcal{D}^i_{train}$ in new sessions

works falls in two main streams, the multi-class incremental learning [2, 13, 15, 26, 30, 49] and the multi-task incremental learning [14, 20, 32]. Early approaches for IL use knowledge distillation [30, 44] to transfer knowledge from the old model to a new model. iCaRL [30] learns a nearest-neighbor classifier with exemplars to preserve performance and combines distillation loss to avoid forgetting. EEIL [2] introduces an end-to-end framework with cross-entropy loss and distillation loss for IL. LUCIR [13] learns a unified classifier to solve the class imbalance problem between the base and new classes. Liu *et al*. [26] propose mnemonics training through bilevel optimizations in model-level and exemplar-level for tackling multi-class incremental learning.

Few-Shot Class-Incremental Learning. FSCIL [1, 31, 39, 56] is recently proposed with the goal of undertaking the CIL task with limited data in incremental sessions. It can also been seen as a few-shot learning task that can classify both novel and old classes at the same time. Tao *et al*. [39] propose a neural gas network to preserve the topology of the features in the base and new classes for the FSCIL task. Ren *et al*. [31] also undertake the few-shot incremental learning task but with a different setting. Our work mainly follows the task definition proposed in [39] which is more closed to the setting in incremental learning literature.

3. Problem Set-up

FSCIL aims to design a machine learning algorithm that can continually learn novel classes from only a few new training examples without forgetting knowledge about old classes. Usually, FSCIL has several learning sessions that come in sequence. Once the learning of model steps into the next session, the training dataset in previous learning sessions are no longer available, while the evaluation of the FSCIL algorithm in each session involves classes in all previous sessions and the current session. To be specific, let $\{\mathcal{D}^0_{train}, \mathcal{D}^1_{train}, \cdots, \mathcal{D}^n_{train}\}$ denotes the training sets of different learning sessions, and the corresponding label space of dataset $\mathcal{D}^i_{train}$ is denoted by $\mathcal{C}^i$. Different datasets have no overlapped classes, *i.e.* $\forall i, j$ and $i \neq j, \mathcal{C}^i \cap \mathcal{C}^j = \varnothing$. At the ith learning session, only $\mathcal{D}^i_{train}$ can be used for network training, and for evaluation, the test dataset $\mathcal{D}^n_{test}$ at session i include test data from all previous and current classes, *i.e.*, the label space of $\mathcal{C}^0 \cup \mathcal{C}^1 \cdots \cup \mathcal{C}^n$. Usually, the training set $\mathcal{D}^0_{train}$ in the first session is a relatively large dataset where a sufficient amount of data is available for training, which is also called the base training set. On the contrary, the datasets in all following sessions have only a limited amount of data, and the dataset $\mathcal{D}^i_{train}$ on a specific session is often described as a $N-$way $K-$shot training set, where there are N classes in the dataset, and each class has K training images. For example, in the popular benchmark dataset CIFAR100, there are 60 classes in the base sessions, and each class has 500 training images, while in each incremental session, only 5 classes are available for training and each class only has 5 images. FSCIL defines a harsh problem setting, where the severe data imbalance and scarcity problems will further exacerbate knowledge forgetting in incremental learning.

4. Method

In this section, we introduce our framework for few-shot incremental learning. We first describe our decoupled training strategy of representations and classifiers in Section 4.1. Then we present our proposed continually evolved classifier in Section 4.2. To enable the learning of CEC, we design a pseudo incremental learning algorithm, which is described in Section 4.3. The overview of the whole training pipeline is shown in Fig. 2.

4.1. Decoupling the Learning of Representations and Classifiers

Our few-shot incremental learning framework mainly includes three training stages: the feature pre-training stage, the pseudo-incremental learning stage, and the classifier learning stage, as shown in Fig. 2. The first two stages use data from the base sessions to learn the network backbone and the classifier adaptation module, and the classifier learning stage only learns the network classifier in each new incoming session.

Feature pre-training stage. It is commonly evidenced in previous incremental learning literature that finetuning the network in new sessions can lead to significant knowledge forgetting of old classes. The data shortage problem in the few-shot incremental learning will further introduce the overfitting problem that exacerbates knowledge forgetting. To tackle this problem, we propose to decouple the learning of representations and classifiers to avoid the catastrophic forgetting issue at incremental stages. Specifically, we first train a convolutional neural network in the standard manner with the training dataset in the base session where abundant data are available for learning image representations, and we can then reuse the network backbone to encode image data in all sessions. By freezing backbone parameters in new sessions, we can avoid knowledge forgetting and overfitting in the representations when learning the model on new sessions.

Pseudo incremental learning stage. Based on the pretrained backbone model, we learn the classifier adaptation module to enable the function of the CEC, which is also based on the base dataset. The adaptation module is frozen after training and is used to to update the classifiers learned on individual sessions. We leave the detailed description of the classifier adaptation module and the training paradigm in Section 4.2 and Section 4.3.

Classifier learning stage. Once the feature backbone and the graph models are learned in the base session, our model can be deployed for incremental learning. We only need to learn a classifier upon the fixed backbone network with the dataset in new sessions, and then the learned classifiers in the current session and previous sessions are fed to the graph model for adaptation. Finally, the updated classifiers can be used for evaluation.

4.2. Continually Evolved Classifier

As image categories come with groups in the incremental learning task, the classifiers learned on individual sessions may only provide discriminative decision boundaries between current classes. When all previous classes are involved for evaluation, the directly concatenated classifiers can not guarantee their discriminative ability and may fail to make correct decisions. Therefore, to derive good decision boundaries over all classes, it is important to ensure the classifier learning incorporates the global context information of all individual tasks in previous sessions. To achieve this goal, we propose a continually evolved classifier which includes a classifier adaptation module to update the classifier weights learned on each individual session based on the global context of previous sessions. Let $\mathbf{W}_i \in \mathbb{R}^{N_i \times C}$ denotes the parameter matrix in the CNN classifier learned on session i, where each row vector $\vec{w}_i^c$ in $\mathbf{W}_i$ is the weights that correspond to a specific class c, N_i is the number of classes in session i and C is the number of feature channels. $\vec{w}_i^c$ can be seen as a prototype vector for class c where the values in different dimensions essentially indicates the discriminability of different channels. To refine the discriminability of the classifier, we can adjust the values in $\vec{w}_i^c$ by looking at $\vec{w}_i^i$ of all other classes. To do so, we first collect the weight vectors of all other classes in the previous sessions:

$$\tilde{\mathbf{W}}_I = \{\vec{w}_0^1, \vec{w}_0^2, ..., \vec{w}_i^1, \vec{w}_i^2, ..., \vec{w}_I^{N_I}\}, \tag{1}$$

where I is the total number of sessions so far. Then, we use the Graph Attention Network (GAT) [41] to model the relations between these prototype vectors and propagate context information, where all the weight vectors in $\tilde{\mathbf{W}}_I$ can be regarded as the nodes in the graph model. The Graph Attention Network has several desirable properties that make it an appropriate tool to encode context information: first, as the updating of graph node is based on attention mechanism, the context encoding is permutation invariant to the sequence of classes during incremental learning. second, the GAT model allows a trained model to be extended to any number of classes, which means that the updating of classifiers at any sessions can share the same learned GAT. Since the nodes are fully connected in the GAT model, it has the similar structure with Transformer [40] that also uses self-attention for information propagation.

To illustrate the context propagation process in the GAT, we take the updating of a node j in the graph as an example. We first compute a relation coefficient e_{jk} between the node j and all nodes in the graph, such as $\vec{w}^j$ and $\vec{w}^k$:

$$e_{jk} = \langle \phi(\vec{w}^j), \theta(\vec{w}^k) \rangle, \tag{2}$$

where, ϕ and θ are linear transformation functions that project the original prototype representations to a new metric space. $\langle \cdot, \cdot \rangle$ is a similarity function that computes the inner product between two vectors. Here we omit the session indexes in the subscript of $\vec{w}^j$ and $\vec{w}^k$ for clarity. We then normalize all the coefficients with the softmax function to get the final attention weights corresponding to the center node j:

$$a_{jk} = \text{softmax}(e_{jk}) = \frac{\exp{(e_{jk})}}{\sum_{h=1}^{|\tilde{\mathbf{W}}|} \exp{(e_{jh})}}. \tag{3}$$

Algorithm 1 Pseudo incremental learning. N_i is the number of classes in pseudo incremental classes; y_q and $\hat{y}_q$ indicates the ground truth label and the network predictions, respectively; $\mathcal{L}(\cdot)$ is the cross-entropy loss function.

```
Input: Base classes datasets $\mathcal{D}^0_{train}$, pre-trained model $\mathcal{R}$,
    a randomly initialized GAT model $\mathcal{G}_\theta$.
Output: A trained GAT model $\mathcal{G}_\theta$.
while not done do
    $\{\mathcal{S}_b, \mathcal{Q}_b\} \leftarrow$ Sample the the support and query set for
    pseudo base classes from $\mathcal{D}^0_{train}$
    $\mathbf{W}_b \leftarrow$ Learn FC layer upon $\mathcal{R}$ with $\mathcal{S}_b$
    $\{\mathcal{S}_i, \mathcal{Q}_i\} \leftarrow$ Sample the support and query set for
    pseudo incremental classes $\mathcal{D}^0_{train}$
    for class $c$ in $N_i$ do
        $\gamma \leftarrow$ Random select angle in $\{90°, 180°, 270°\}$;
        $\{\mathcal{S}'_i, \mathcal{Q}'_i\} \leftarrow$ Rotate $\{\mathcal{S}_i, \mathcal{Q}_i\}$ from class $c$ with the
        selected angle $\gamma$;
    end for
    $\mathbf{W}_i \leftarrow$ Learn FC layer upon $\mathcal{R}$ with pseudo incre-
    mental support set $\mathcal{S}'_i$ after rotation
    $\{\mathbf{W}'_b, \mathbf{W}'_i\} \leftarrow$ Update classifier $\{\mathbf{W}_b, \mathbf{W}_i\}$ using $\mathcal{G}_\theta$

    $\hat{y}_q \leftarrow$ Make predictions for $\{\mathcal{Q}_b, \mathcal{Q}'_i\}$ using
    $[\mathcal{R}, (\mathbf{W}'_b, \mathbf{W}'_i)]$
    loss $\leftarrow$ Compute loss with $\mathcal{L}(y_q, \hat{y}_q)$,
    Optimize $\mathcal{G}_\theta$ with SGD
end while
```

Based on the normalized attention coefficients a_{jk}, we aggregate information from all the nodes in the graph based on a_{jk} and fuse it with the original node representation to obtain $\vec{w}^{j\prime}$:

$$\vec{w}^{j\prime} = \vec{w}^j + \Big(\sum_{k=1}^{|\tilde{\mathbf{W}}_I|} a_{jk}\mathbf{U}\vec{w}^k\Big), \tag{4}$$

where $\mathbf{U}$ is the weight matrix of a linear transformation. We repeat the operations above to update the embeddings of all nodes in the graph, and finally we obtain the updated classifiers:

$$\tilde{\mathbf{W}_I}' = \{\vec{w}_0^{1\prime}, \vec{w}_0^{2\prime}, ..., \vec{w}_i^{1\prime}, \vec{w}_i^{2\prime}, ..., \vec{w}_I^{N_I\prime}\}, \tag{5}$$

In each incoming session, we use the adaptation module to update the classifiers learned in the current session and previous sessions, and then concatenate the updated classifiers to make predictions over all classes. Many useful practices can be adopted to improve the knowledge propagation, such as multi-head attention [40, 41], layer normalization [40], and dropout [40]. We also follow [47] that incorporates the embedding of the network input into the graph to help the learning of context knowledge.

4.3. Pseudo Incremental Learning

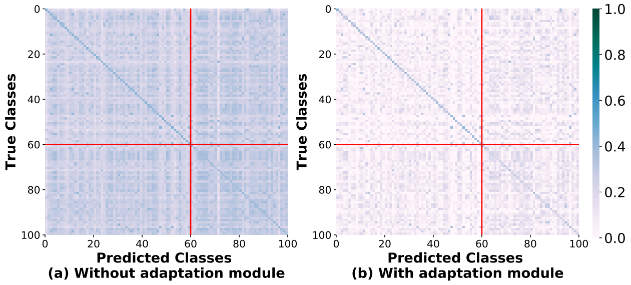
In order to enforce context encoding in the classifier adaptation module, it is important to learn the GAT under the incremental learning scenario. However, in FSCIL, only data from a single session are available for training, and the amount of data in incremental sessions is always limited. To overcome this problem, we design a pseudo incremental learning algorithm to train the adaptation module by episodically constructing pseudo incremental tasks based on the base dataset $\mathcal{D}^0_{train}$ to mimic the test scenario. The pseudo code of the proposed algorithm is illustrated in Alg. 1. Our algorithm takes inspirations from meta-learning [42], where a small classification task is constructed to enable learning on the meta-level beyond a specific task. We utilize data from the base dataset $\mathcal{D}^0_{train}$ to construct small incremental learning tasks for network training, where some sampled classes play the role of the base classes in incremental learning, while the other classes play the role of the incremental classes. Specifically, both pseudo incremental classes and pseudo base classes have the support set and the query set, which are denoted by $(\mathcal{S}_b, \mathcal{Q}_b)$ and $(\mathcal{S}_i, \mathcal{Q}_i)$, respectively. The support set is used to learn the classifier weights of different classes, and the query set is used to compute loss for optimization. To be concrete, we first use the support sets ($\mathcal{S}_b$ and $\mathcal{S}_i$) to learn two classifiers, ($\mathbf{W}'_b$ and $\mathbf{W}'_i$), for pseudo base classes and pseudo incremental classes respectively. Then, the two classifiers are concatenated and fed into the adaptation module $\mathcal{G}_\theta$ for updating. We use the updated classifiers $(\mathbf{W}'_b, \mathbf{W}'_i)$ to make predictions for the query sets of pseudo base classes and pseudo incremental classes, *i.e.*, $\mathcal{Q}_b$ and $\mathcal{Q}_i$, and compute the loss to optimize the adaptation module $\mathcal{G}_\theta$. We also finetune the last layer of the backbone with a small learning rate during PIL, which we find helpful. In our experiment, we find that directly splitting the sampled base classes into two groups to train the adaptation module fails. A possible reason is that the backbone model pre-trained on base classes can well separate these sampled classes already without context information. As a result, the training may simply bypass the adaptation module. To handle this issue, we randomly rotate the data of the sampled pseudo incremental classes, $(\mathcal{S}_i, \mathcal{Q}_i)$, with a large class-wise angle γ to synthesis new classes, as we observe that rotating data with a large angle can make the synthesized images lose parts of the semantics of their original classes, but demonstrate similar semantics among synthesized images. Once the adaptation module is learned, we can freeze the parameters in the adaptation module and deploy it in the new incremental sessions.

5. Experiments

In this section, we evaluate our proposed CEC on three popular few-shot incremental learning benchmark datasets,

Method	Decoupled	AM	PIL	Acc. in each session (%) ↑											PD ↓
				0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
Linear				71.02	2.70	0.63	0.82	0.76	0.71	0.66	0.62	0.59	0.56	0.53	70.49
Linear	✓			71.02	7.22	5.25	3.59	6.13	6.46	7.73	5.87	4.55	3.88	3.92	67.10
Cosine				73.32	35.53	14.64	1.47	0.73	0.68	2.45	0.60	0.57	0.55	0.52	72.80
Cosine	✓			74.36	48.50	43.40	39.26	37.29	33.69	33.36	32.49	31.81	31.19	30.36	44.00
Linear+Data Init.	✓			66.58	59.38	54.29	50.00	46.34	43.18	40.43	38.00	35.85	34.55	32.76	33.83
L2+Data Init.	✓			67.75	59.11	56.05	51.75	51.39	47.19	46.97	45.01	42.77	42.94	41.62	26.13
DeepEMD+Data Init.	✓			75.35	70.69	66.68	62.34	59.76	56.54	54.61	52.52	50.73	49.20	47.60	27.75
Cosine+Data Init.	✓			75.52	70.95	66.46	61.20	60.86	56.88	55.40	53.49	51.94	50.93	49.31	26.21
Cosine+Data Init.	✓	✓		75.60	71.00	66.89	61.81	60.86	56.81	56.11	53.59	52.52	50.59	49.15	26.45
Cosine+Data Init.	✓	✓	✓	**75.85**	**71.94**	**68.50**	**63.50**	**62.43**	**58.27**	**57.73**	**55.81**	**54.83**	**53.52**	**52.28**	**23.57**

Table 1: Ablation study on CUB200 to analyze the effectiveness of different components in our model. **AM** is the adaptation module, **PIL** is pseudo incremental learning, and **PD** denotes the performance dropping rate.



(a) Without adaptation module (b) With adaptation module

Figure 3: Confusion matrices with and without adaptation module on CIFAR100. We use red lines to separate regions of base classes and incremental classes. Our adaptation module effectively improves the network prediction, which results in a less scattered confusion matrix.

including CIFAR100 [18], *mini*ImageNet [33] and Caltech-UCSD Birds-200-2011 (CUB200) [43]. We first present the experiment details and dataset statistics. Then we conduct comprehensive experiments to validate the the effectiveness of individual components in our design and study their characteristics. Finally, we compare our network with state-of-the-art methods on the benchmarks.

5.1. Dataset

CIFAR100. CIFAR100 is a classification dataset with 60,000 32 × 32 RGB images from 100 classes. Each class contains 500 training images and 100 testing images. We follow the splits in [39], where 60 classes and 40 classes are used as base classes and new classes, respectively. The 40 new classes are further divided into 8 new incremental sessions, and each new session is a 5-way 5-shot classification task.

***mini*ImageNet.** *mini*ImageNet contains 100 classes with 600 images in each class, which are built upon the ImageNet dataset [33]. The image size of *mini*ImageNet is 84×84 and we follow [39] to split the 100 classes into 60 base classes and 40 incremental classes. The 40 new classes are further divided equally into 8 sessions with 5 classes in each session, and each class has 5 training images in the incremental sessions.

Caltech-UCSD Birds-200-2011. CUB200 [43] was originally proposed for fine-grained image classification. It contains 11,788 images from 200 classes. We follow the splits in [39] that 200 classes are divided into 100 base classes and 100 new classes, respectively. The 100 new classes are further divided into 10 new sessions where each session is a 10-way 5-shot task. The images size in CUB200 is 224 × 224.

5.2. Implementation Details

Following [39], we employ ResNet20 [10] as the backbone for experiments on CIFAR100 and ResNet18 [10] for experiments on miniImageNet and CUB200. Our network is built with PyTorch library, and we use SGD with momentum for optimization. At the pseudo incremental learning stage, we random choose the angle γ from $\{90^\circ, 180^\circ, 270^\circ\}$ to synthesize new classes. We train the graph model $\mathcal{G}_\theta$ for 5000 iterations with the learning rate of 0.0002. The learning rate is decayed by 0.5 every 1000 iteration. Random crop, random scale, and random horizontal flip are used for data augmentation at training time.

Evaluation Protocol. We evaluate the model after each session with the test set $\mathcal{D}^i_{test}$ and report the Top 1 accuracy. We also define a performance dropping rate (**PD**) that measures the absolute accuracy drops in the last session w.r.t. the accuracy in the first session, *i.e.*, PD $= \mathcal{A}_0 - \mathcal{A}_N$, where $\mathcal{A}_0$ is the classification accuracy in the base session and $\mathcal{A}_N$ is the accuracy in the last session.

5.3. Analysis

In this part, we implement various experiments to evaluate the effectiveness of our algorithm and study the characteristics of different components. For analysis, we mainly report the results on the CUB200 dataset and leave other datasets in Section 5.4 and our supplementary material.

Ablation study. In the beginning, we conduct an ablative analysis on the CUB200 dataset to observe the effectiveness of the different components in our model. We first consider four kinds of classifiers, including the vanilla **linear** classifier in the CNNs, the **cosine classifier** [42], the **L2** classifier [35], and the **DeepEMD** classifier [52], where

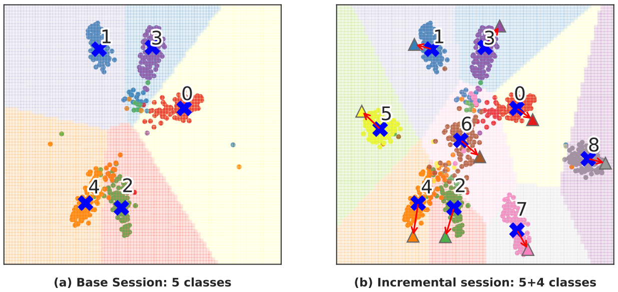


Figure 4: t-SNE [27] visualization of data embeddings and classifier weights before and after the adaptation module. Dots with different colors represent data points from different classes. The blue crosses indicate the classifier weights before adaptation. Triangles indicate the weights after adaptation. Red arrows show the changes in weights caused by the adaptation module. Our adaptation module moves the classifier weights away from the confusion area and generates better decision boundaries.

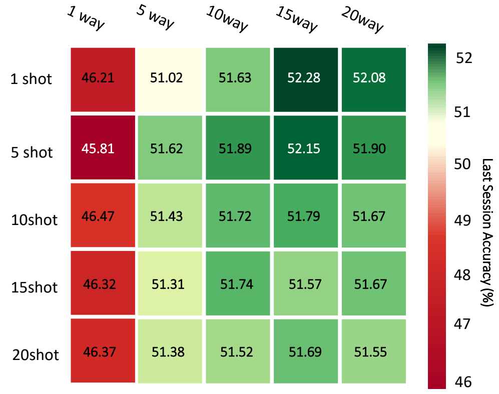


Figure 5: Comparison of different ways and shots for pseudo incremental learning. We report the accuracy of the last incremental session on the CUB200 [43] dataset for comparison. Large ways and low shots are preferred for learning of the adaptation module.

their main difference is the metric to compute class scores given the prototypes of each class. In new incremental sessions, the classifier is learned with a learning rate of 0.1 for 100 epochs. We also try using the data embeddings to parameterize the classifier weights where the weight vector of each class is initialized by the average data embeddings in the training set, which is denoted by **Data Init**. We gradually involve our designs to observe their influence on performance, including decoupled training scheme (**Decoupled**), the adaptation module (**AM**), and the pseudo incremental learning paradigm (**PIL**). When our adaptation module is not trained with pseudo incremental learning, we adopt the meta-learning [42] to learn the parameters in the graph. The result is shown in Table 1. For both cosine classifier and the linear classifier, decoupling the representation learning and the classifier learning is useful for avoiding the catastrophic forgetting issue, which can decrease the performance dropping rate by 28.81% and 3.39%, respectively. Using the data embeddings to initialize the classifier weights is beneficial to all classifiers. When both the decoupled training strategy and data initialization are adopted, all four classifiers can achieve good performance, and cosine classifier performs the best. Without further specification, we use the cosine classifier in rest experiments. Using meta-learning to learn the adaptation module fails to improve the performance. When the adaptation module is learned by our proposed PIL, it can boost the performance over all sessions by up to 3.13% and can decrease performance dropping rate by 2.87%.

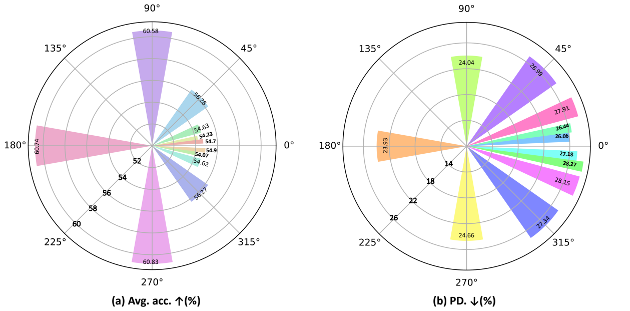


Figure 6: Comparison of different rotation degrees for pseudo incremental learning. Our tested degrees include $180°$, $\pm 90°$, $\pm 45°$, $\pm 20°$, $\pm 10°$ and $\pm 5°$. We report the average accuracy of all the sessions and the performance dropping (PD) on the CUB200 dataset for comparison. Large rotation degrees, such as $180°$ and $\pm 90°$, are more helpful for class synthesis.

Confusion Matrix. To further observe the behavior in the adaptation module, we plot the confusion matrix generated by the models with and without our adaptation module in Fig. 3. As we can see, the classifier without adaptation generates a confusing matrix, particularly for the incremental classes (the prediction distribution is more scattered and thus darker). In contrast, our adaptation module can effectively improve the predictions where the values more lie in the diagonal that indicates the ground truth.

Visualization of adaptation. We plot the data embeddings and classifier weights in low-dimension space with t-SNE [27] in Fig. 4. We randomly choose five classes from the CIFAR100 dataset as the base classes, and we add four new classes as the incremental classes. As can be seen, the adaptation module moves the classifier weights away from the confusion area to generate better decision boundaries when new classes are involved.

Analysis of pseudo incremental learning. We next investigate the configurations in the pseudo incremental learning scheme. In particular, we fix the query number as 10 and analyze the influence of ways, shots and the rotation angles during pseudo incremental learning. We set the same ways, shots and queries for pseudo base classes and pseudo incremental classes. The comparison is shown in Fig. 5. We choose the number of ways from $\{1, 5, 10, 15, 20\}$ and the number of shots from $\{1, 5, 10, 15, 20\}$. We find that a rel-

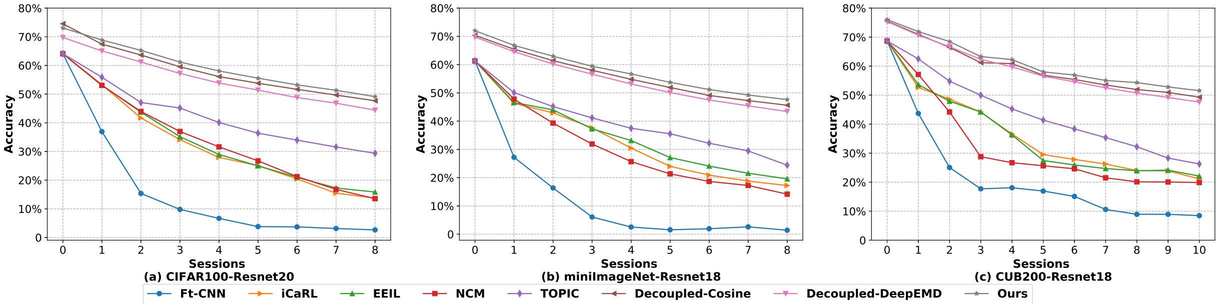


Figure 7: Comparison with the state-of-the-art on three benchmarks: (a) CIFAR100 (b) miniImageNet and (c) CUB200. Our method outperforms previous works with significant performance advantages. Please refer to Table 2 and our supplementary material for detailed numbers.

Method	Acc. in each session (%) ↑											PD ↓	our relative improvement
	0	1	2	3	4	5	6	7	8	9	10		
Ft-CNN	68.68	43.7	25.05	17.72	18.08	16.95	15.1	10.6	8.93	8.93	8.47	60.21	**+36.64**
iCaRL* [30]	68.68	52.65	48.61	44.16	36.62	29.52	27.83	26.26	24.01	23.89	21.16	47.52	**+23.95**
EEIL* [2]	68.68	53.63	47.91	44.2	36.3	27.46	25.93	24.7	23.95	24.13	22.11	46.57	**+23.00**
NCM* [13]	68.68	57.12	44.21	28.78	26.71	25.66	24.62	21.52	20.12	20.06	19.87	48.81	**+25.24**
TOPIC [39]	68.68	62.49	54.81	49.99	45.25	41.4	38.35	35.36	32.22	28.31	26.28	42.40	**+18.83**
Decoupled-Cosine [42]‡	75.52	70.95	66.46	61.20	60.86	56.88	55.40	53.49	51.94	50.93	49.31	26.21	**+2.64**
Decoupled-DeepEMD [52]‡	75.35	70.69	66.68	62.34	59.76	56.54	54.61	52.52	50.73	49.20	47.60	27.75	**+4.18**
CEC (Ours)	**75.85**	**71.94**	**68.50**	**63.5**	**62.43**	**58.27**	**57.73**	**55.81**	**54.83**	**53.52**	**52.28**	**23.57**	

‡ Our implementation.

Table 2: Comparison with the state-of-the-art on CUB200 dataset. * indicates results copied from TOPIC [39]. Please refer to our supplementary material for the detailed results on other datasets.

atively larger way and a smaller shot are better, and the optimal result is obtained when the way is 15 and the shot is 1.

We then fix the way and shot, and investigate the rotation degrees for classes synthesis in PIL. We choose different rotation degrees for comparison and present their results in Fig. 6. Our tested degrees include $180°$, $\pm 90°$, $\pm 45°$, $\pm 20°$, $\pm 10°$ and $\pm 5°$. As we can see, large angles, such as $180°$, $90°$ and $-90°$ ($270°$) are more effective for class synthesis and generate higher average accuracy and lower performance dropping rate. When the rotation degree is small, the synthesized classes may be confused with the original classes and thus generate poor results. When the three large degrees, *i.e.*, $\{180°,\ 90°,\ -90°(270°)\ \}$ are randomly selected for training, it generates the best result with the highest average accuracy of 61.33% and lowest dropping rate of 23.57%.

5.4. Comparison with the State-of-the-Art Methods

Finally, we compare our performance with the state-of-the-art results on three benchmarks: CIFAR100, miniImagenet, and CUB200. We show the results in Fig. 7 and the detailed numbers for CUB200 in Table 2 (Please refer to our supplementary material for results on other datasets). Our model has the highest average accuracy over all sessions and the lowest performance dropping rate. Particularly, our PD outperforms the state-of-the-art results by 10.80% on CIFAR100, 12.52% on *mini*ImageNet and 18.83% on CUB200.

6. Conclusion

In this paper, we solve the few-shot incremental learning problems from two aspects. We first adopt a decoupled learning strategy to separate the learning of representations and classifiers, which effectively avoid knowledge forgetting in the backbone. Then, we propose a continually evolved classifier for few-shot incremental learning, which employs an adaptation module to update the classifier weights based on a global context of all sessions. To enable the learning of the adaptation module, we propose a pseudo incremental learning paradigm. Experiments on three datasets show that our method significantly outperforms the baselines and the state-of-the-art approaches.

Acknowledgement

This work was supported by Alibaba Group through Alibaba Innovative Research (AIR) Program and Alibaba-NTU Singaproe Joint Research Institute (JRI), Nanyang Technological University, Singapore. This research is also supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2018-003), and the MOE Tier-1 research grants: RG28/18 (S), RG22/19 (S) and RG95/20.

References

[1] Ali Ayub and Alan R Wagner. Cognitively-inspired model for incremental learning using a few examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 222–223, 2020. 3

[2] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018. 3, 8

[3] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *ICLR*, 2019. 2

[4] Hung-Jen Chen, An-Chieh Cheng, Da-Cheng Juan, Wei Wei, and Min Sun. Mitigating forgetting in online continual learning via instance-aware parameterization. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[5] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *International Conference on Learning Representations*, 2019. 2

[6] Xiaoyu Chen, Chi Zhang, Guosheng Lin, and Jing Han. Compositional prototype network with multi-view comparision for few-shot point cloud semantic segmentation. *arXiv preprint arXiv:2012.14255*, 2020. 2

[7] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5138–5146, 2019. 2

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 2

[9] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375, 2018. 2

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 6

[11] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *arXiv*, 1503.02531, 2015. 1

[12] Ruibing Hou, Hong Chang, MA Bingpeng, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. In *Advances in Neural Information Processing Systems*, pages 4003–4014, 2019. 2

[13] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019. 3, 8

[14] Wenpeng Hu, Zhou Lin, Bing Liu, Chongyang Tao, Zhengwei Tao, Jinwen Ma, Dongyan Zhao, and Rui Yan. Overcoming catastrophic forgetting via model adaptation. In *ICLR*, 2019. 3

[15] Xinting Hu, Kaihua Tang, Chunyan Miao, Xian-Sheng Hua, and Hanwang Zhang. Distilling causal effect of data in class-incremental learning. *arXiv preprint arXiv:2103.01737*, 2021. 3

[16] Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[17] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 1

[18] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 2009. 6

[19] Guankai Li, Chi Zhang, and Guosheng Lin. Cyclesegnet: Object co-segmentation with cycle refinement and region correspondence. *arXiv preprint arXiv:2101.01308*, 2021. 1

[20] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 3

[21] Binghao Liu, Yao Ding, Jianbin Jiao, Ji Xiangyang, and Qixiang Ye. Anti-aliasing semantic reconstruction for few-shot semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 2

[22] Weide Liu, Chi Zhang, Guosheng Lin, Tzu-Yi HUNG, and Chunyan Miao. Weakly supervised segmentation with maximum bipartite graph matching. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2085–2094, 2020. 1

[23] Weide Liu, Chi Zhang, Guosheng Lin, and Fayao Liu. Crnet: Cross-reference networks for few-shot segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4165–4173, 2020. 2

[24] Yaoyao Liu, Bernt Schiele, and Qianru Sun. An ensemble of epoch-wise empirical bayes for few-shot learning. In *European Conference on Computer Vision*, pages 404–421. Springer, 2020. 2

[25] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[26] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[27] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 7

[28] Eunbyung Park and Junier B Oliva. Meta-curvature. In *Advances in Neural Information Processing Systems*, pages 3309–3319, 2019. 2

[29] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 2

[30] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: incremental classifier and representation learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 3, 8

[31] Mengye Ren, Renjie Liao, Ethan Fetaya, and Richard S. Zemel. Incremental few-shot learning with attention attractor networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3

[32] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *ICLR*, 2019. 3

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 6

[34] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *NIPS*, pages 2990–2999, 2017. 1

[35] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017. 2, 6

[36] Qianru Sun, Yaoyao Liu, Zhaozheng Chen, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning through hard tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2

[37] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *CVPR*, 2019. 2

[38] Xin Sun, Zhenning Yang, Chi Zhang, Keck-Voon Ling, and Guohao Peng. Conditional gaussian distribution learning for open set recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13480–13489, 2020. 1

[39] Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 1, 3, 6, 8

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 4, 5

[41] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. 2, 4, 5

[42] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NIPS*, 2016. 2, 5, 6, 7, 8

[43] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 6, 7

[44] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019. 3

[45] Ze Yang, Yali Wang, Xianyu Chen, Jianzhuang Liu, and Yu Qiao. Context-transformer: tackling object confusion for few-shot detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. 2

[46] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Learning adaptive classifiers synthesis for generalized few-shot learning. *arXiv preprint arXiv:1906.02944*, 2019. 2

[47] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 5

[48] Han-Jia Ye, De-Chuan Zhan, Yuan Jiang, and Zhi-Hua Zhou. Heterogeneous few-shot model rectification with semantic mapping. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2

[49] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6982–6991, 2020. 3

[50] Zhongqi Yue, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. Interventional few-shot learning. In *NeurIPS*, 2020. 2

[51] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Differentiable earth mover's distance for few-shot learning. *arXiv e-prints*, 2020. 2

[52] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 6, 8

[53] Chi Zhang, Guosheng Lin, Fayao Liu, Jiushuang Guo, Qingyao Wu, and Rui Yao. Pyramid graph networks with connection attentions for region-based one-shot semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9587–9595, 2019. 2

[54] Chi Zhang, Guosheng Lin, Fayao Liu, Rui Yao, and Chunhua Shen. Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5217–5226, 2019. 2

[55] Chi Zhang, Rui Yao, and Jinpeng Cai. Efficient eye typing with 9-direction gaze estimation. *Multimedia Tools and Applications*, 77(15):19679–19696, 2018. 1

[56] Hanbin Zhao, Yongjian Fu, Xuewei Li, Songyuan Li, Bourahla Omar, and Xi Li. Few-shot class-incremental learning via feature space composition. *arXiv preprint arXiv:2006.15524*, 2020. 3