Pattern Recognition 128 (2022) 108662

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Learning multi-level weight-centric features for few-shot learning☆

Mingjiang Liang[a], Shaoli Huang[b,*], Shirui Pan[c], Mingming Gong[d], Wei Liu[a,*]

[a] University of Technology Sydney, Australia
[b] Tencent AI Lab, China
[c] Monash University, Australia
[d] University of Melbourne, Australia

ARTICLE INFO

Article history:
Received 4 May 2021
Revised 26 February 2022
Accepted 19 March 2022
Available online 24 March 2022

Keywords:
Fewshot learning
Low-shot learning
Multi-level features
Image classification

ABSTRACT

Few-shot learning is currently enjoying a considerable resurgence of interest, aided by the recent advance of deep learning. Contemporary approaches based on weight-generation scheme delivers a straightforward and flexible solution to the problem. However, they did not fully consider both the representation power for unseen categories and weight generation capacity in feature learning, making it a significant performance bottleneck. This paper proposes a multi-level weight-centric feature learning to give full play to feature extractor's dual roles in few-shot learning. Our proposed method consists of two essential techniques: a weight-centric training strategy to improve the features' prototype-ability and a multi-level feature incorporating a mid- and relation-level information. The former increases the feasibility of constructing a discriminative decision boundary based on a few samples. Simultaneously, the latter helps improve the transferability for characterizing novel classes and preserve classification capability for base classes. We extensively evaluate our approach to low-shot classification benchmarks. Experiments demonstrate our proposed method significantly outperforms its counterparts in both standard and generalized settings and using different network backbones.

1. Introduction

Despite remarkable success made in visual recognition tasks [1–4], deep learning models generally lack versatility and extendability, hindering their applicability in practice. For instance, being data-hungry to learn massive parameters, deep neural networks often fail to work well in data-scarce environments [5,6]. Besides, a trained model's prediction domain is usually not expandable unless re-executing the training process. In response to these deficiencies, there has been increasing efforts devoted to few-shot learning (FSL) [7–11]. Moreover, the exploration of FSL is gradually expanding in various research problems such as FKP recognition [12], Medical image classification [13], object detection [14], text classification [15], and instance credibility inference [16].

FSL refers to a technique that exploits knowledge from base-class data (provided auxiliary training set) to allow models to understand new concepts from only a few examples [15,17–20]. Existing approaches to this problem mainly consist of meta-learning and weight-generation based frameworks. The former focuses on learning a meta-learner from base-class data to facilitate learning a new-task learner. Although meta-learning approaches achieve great success, they often require sophisticated training procedures and are difficult to extend to generalized few-shot learning (GFSL) settings. By contrast, the weight-generation framework delivers a more straightforward and flexible solution. This type of method first learns an embedding space from base-class data and then utilizes the embedding of support set (novel-class training samples) to construct the corresponding classifier weights(as illustrated in Fig. 1. This learning regime simplifies the few-shot learning problem by mainly focusing on feature learning and weight generation, enabling a trained model's extendability.

In the framework, feature learning is crucial due to its dual-use mechanism (representing images and constructing classifiers). However, existing methods learn a feature extractor without considering three essential issues associated with its dual-functionality: representation transferability, base-class memorability, and prototype-ability. The transferability refers to whether the learned representation from base-class data is transferable to the novel-class data. Recent works [10,21–23] mainly extract features from the last Conv layer of deep models, leading to less transferability of the feature extractor. This can be attributed to

☆ This research is supported by an Australian Government Research Training Program Scholarship.

* Corresponding authors.
E-mail addresses: mingjiang.liang@student.uts.edu.au (M. Liang), shaolihuang@tencent.com (S. Huang), shirui.pan@monash.edu (S. Pan), mingming.gong@unimelb.edu.au (M. Gong), wei.liu@uts.edu.au (W. Liu).

https://doi.org/10.1016/j.patcog.2022.108662
0031-3203/

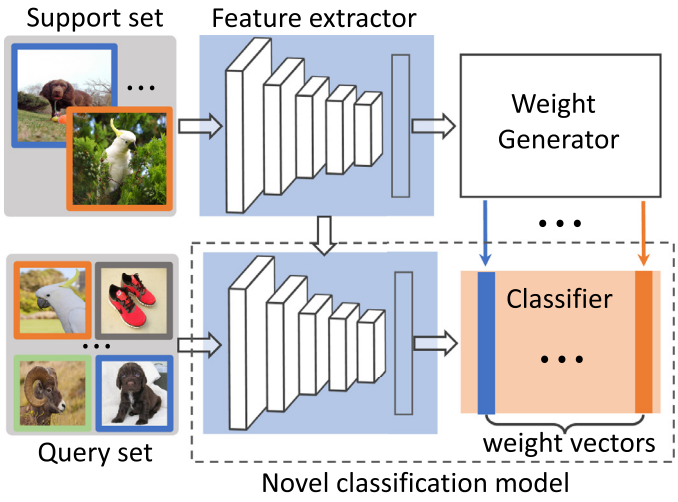

Fig. 1. A general framework of weight generation methods for few-shot learning. Learning a good feature extractor plays a vital role in this framework, as it is used for novel models to extract image features and generate classifier weights for new categories.

the fact that higher layer activations with higher specialization to base-class task are less transferable to novel-class task [24] when there is a large domain gap between the two tasks. The memorability and prototype-ability are more related to the quality of the generated classifier weights. A GFSL model requires preserving the classification performance for the base-class data. This requirement necessitates base-class memorability to prevent novel-class weights from classifying base-class data to novel classes. The prototype-ability refers to the feature extractor's capacity in allowing few-shot examples to approximate their class-specific prototype. Current methods attempt to complement these two capabilities by learning a weight generation network. Nevertheless, similar to meta-learning approaches, they need training a new task-specific learner for weight generation, limiting the flexibility to construct few-shot classification models. Besides, the weight generator learns the required information from extracted features but fails to access more information through the feature learning stage. To sum up, the existing weight-generation methods do not fully consider the feature extractor's dual-capacity in FSL, which may be a bottleneck of performance.

In this paper, we propose a multi-level weight-centric feature extractor to complement the capacity of current weight-generation methods. We first introduce a weight-centric training strategy to increase the possibility that each sample can approximate its category prototype. Specifically, we fix the classifier weights in the latter learning stage and then enforce samples closer to their corresponding classifier weight in the embedding space. Besides, we build the multi-level feature by incorporating a mid-level and relation-level learning branch with high-level feature learning. The mid-level learning branch extracts mid-level features from intermediate layers while the relation-level one obtains category-relation information from softening predictions. We finally integrate the multi-level information extraction and the weigh-centric strategy into an overall feature learning framework.

Our proposed method ensures the feature extractor's comprehensiveness for advancing few-shot learning. On the one hand, the weight-centric strategy reduces the intra-class variance, improving feature representation generalization. It also pushes data points that are closer to the hyperplane far way. This effect indirectly achieves larger margin classification-boundaries, increasing the feasibility of constructing a discriminative decision boundary based on a few samples. On the other hand, the mid-level features are more transferable [25] to novel classes, and the relation-level representation exhibits higher-level abstraction and more specific to base categories. Therefore, by jointly representing images using these two additional information sources, the resulting model has higher transferability for characterizing novel classes and better preserves classification capability for base classes.

We extensively evaluate our approach on two low-shot classification benchmarks in both standard and generalized FSL learning settings. Experiments show that our proposed method significantly outperforms its counterparts in both learning settings and using different network backbones. We also demonstrate that the mid-level features exhibit strong transferability even in a cross-task environment and the relation-level features help preserve base-class accuracy in the generalized FSL setting.

The contribution of this paper can be summarized as :

- We propose a weight-centric learning strategy that helps reduce the intra-class variance of novel-class data.
- we propose a multi-level feature learning framework, which demonstrates its strong prototype-ability and transferability even in a cross-task environment for few-shot learning.
- We extensively evaluate our approach on two low-shot classification benchmarks in both standard and generalized FSL learning settings. Our results show that the mid-level features exhibit strong transferability even in a cross-task environment while the relation-level features help preserve base-class accuracy in the generalized FSL setting

2. Related work

Recently proposed approaches to few-shot learning problem can be roughly divided into meta-learning based [26–31] and weight-generation based approaches [21–23,32,33].

Meta-learning based methods tackle the few-shot learning problem by training a meta-learner to help a learner can effectively learn a new task on very few training data [27,28,34–37]. Most of these methods are normally designed based on some standard practices for training deep models on limited data, such as finding good weights initialization [27] or performing data augmentation [28] to prevent overfitting. For instance, Finnn et al. [27] propose to learn a set of parameters to initialize the learner model so that it can be quickly adapted to a new task with only a few gradient descent steps [28]; deal with the data deficiency in a more straightforward way, in which a generator is trained on meta-training data and used to augment feature of novel examples for training the learner. Another line of work addresses the problem in a "learning-to-optimize" way [29,36]. For example, Ravi et al. [29] train an LSTM-based meta-learner as an optimizer to update the learner and store the previous update records into the external memory. Though this group of methods achieves promising results, they either require to design complex inference mechanisms [38] or to further train a classifier for novel concepts [27,29]. Our work focuses on learning a feature extractor with dual functions (ie feature representation and classifier weight generation) for FSL problems. Therefore, the major difference from meta-learning techniques is that our method only needs to learn a base model and can construct new models directly using sample features.

Weight-genration based approaches mainly learn an embedding space, in which images are easy to classify using a distance-based classifier such as cosine similarity or nearest neighbor. To do so, Koch et al. [32] trains a Siamese network that learns a metric space to perform comparisons between images. Vinyals et al. [23] propose Matching Networks to learn a contextual embedding,with which the label of a test example can be predicted by looking for its nearest neighbors from the support set. Prototypical networks [39] determine the class label of a test example by

measuring the distance from all the class means of the support set. Since the distance functions of these two works are predefined, [40] further introduce a learnable distance metric for comparing query and support samples. Ji et al. [41] propose a reweighting mechanism to improve the instance representativeness and an information-guidance mechanism to encode discriminative knowledge. Guo and Cheung [42] presents an Attentive Weights Generation via Information Maximization strategy that generates optimal classification weights for each query sample within the task by self-attention and cross-attention paths.

The most related methods to ours are [21,22,43]. These approaches learn a feature representation by a cosine softmax loss, allowing a few novel examples to construct the classifier. Our proposed method differs from them in two folds. First, they only learn a single level of representation, resulting in a limited representation capability, while ours constructs a multi-level model that considers multiple knowledge sources. Furthermore, those methods do not explicitly consider the prototype-ability (the ability to approximate the corresponding prototype by one or several sample features) in learning the feature extractor. In contrast, we introduce a weight-centric learning strategy that makes it more feasible to construct classifier weights from a few samples.

2.1. Analyzing the transferability of ConvNets

Deep learning models are quite data-hungry but nonetheless transfer learning have been proven highly effective to avoid overfitting when training larger models on smaller datasets [44–46]. These findings raise interest in studying the transferability of deep models features in recent years. Yosinski et al. [24] experimentally show how transferable of each layer by quantifying the generality versus specificity of its features from a deep ConvNet, and suggest that higher layer activations with higher specialization to source tasks are less transferable to target tasks. Pulkit et al. [47] investigates several aspects that impact the performance of ConvNet models for object recognition. Hossein et al. [48] identifies several factors that affect the transferability of ConvNet features and demonstrates optimizing these factors aid transferring task. However, these works mainly explore the transferability and generalization ability of ConvNet features in terms of target datasets where the training samples are much more than the few-shot setting. In this work, we investigate the capacities of the intermediate layer, last feature layer, and softmax logits to perform few-shot learning tasks.

3. Methodology

In this section, we first introduce some general notation used throughout the paper. We first briefly review a general weight-generation-based framework for few-shot learning. We further introduce our method for learning base models. Finally, we describe how to utilize these base models in few-shot learning.

3.1. Notation

Let $f_\Theta(\cdot) \in \mathbb{R}^d$ be a feature extractor parameterized by Θ and $W \in \mathbb{R}^{d\times c}$ be a weight matrix of a linear classifier. Here, d is the dimension of the output feature and c is the number of labels for the classification task. We further define $M(\cdot)$ as a neural network classification model, such that $M(f_\Theta(x), W) = W^T f_\Theta(x)$ given an input image x. We denote the training set D_{train} and the test set D_{test}. Slightly different from the general classification setting, few-shot learning train a model $M(\cdot)$ on the training data that consists of a base- and novel-class dataset, that is $D_{train} = D^b_{train} \cup D^n_{train}$. Here, $D^b_{train} = \{(x_i, y_i), y_i \in Y^b\}_{i=1}^{N^b}$ is an abundant dataset while $D^n_{train} = \{(x_i, y_i), y_i \in Y^n\}_{i=1}^{N^n}$ contains very few samples for each label; Y^b and Y^n refers to two different label spaces and $Y^b \cap Y^n = \emptyset$. We further denote the weight matrices W^b and W^n which are corresponding to Y^b and Y^n respectively.

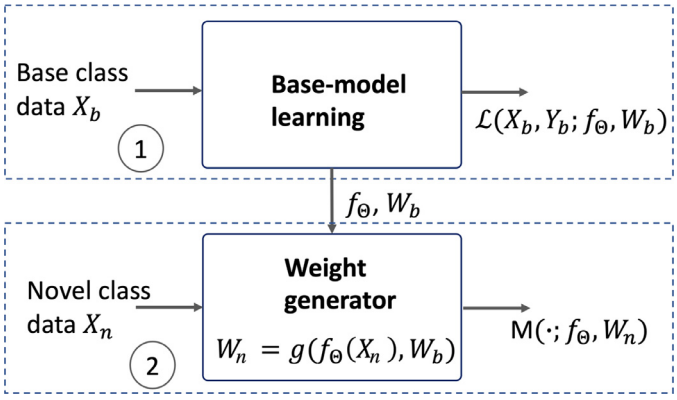

Fig. 2. A general weight-generation-based framework for few-shot learning. Here, $\mathcal{L}$ is the loss function for learning a base model on base-class data. f_Θ and W_b are the feature extractor and classifier weights of the base model. $g(\cdot)$ is weight generator which can be defined or learned from data. $M(\cdot)$ is novel model built for novel categories.

3.2. Weight-generation-based framework

Weight-generation-based approaches have gained increasing attention in recent years, due to its simplicity and flexibility. The general framework for these methods usually consists of two stages: base-model learning and weight generation. As shown in Fig. 2, this framework first learns a classification base-model on base-class dataset. In the second stage, based on the feature extractor $f_\Theta(\cdot)$ and classifier weights W^b of the base-model, a weight generator $g_\phi(.)$ is used to infer the weight vector w given training set $X^y = \{x^y_1, \ldots, x^y_k\}$. Here, the label y is in an unseen label space Y^n and k is usually a small number. In recent literature, there are two typical weight generators : average-based $w_{avg} = g^{avg}(f_\Theta(X^y))$ and attention-based $w_{att} = g^{att}_\phi(f_\Theta(X^y), W^b)$. The former simply compute the mean of the normalized features of training samples, which is expressed as:

$$w_{avg} = \frac{1}{k}\sum_{i=1}^{k} z_i, \tag{1}$$

where z_i is a L_2 norm of the feature vector $f_\Theta(x^y_i)$.

The second one employs an attention-based mechanism to exploit both the sample features and the base-class weights in generating the novel-class weights. The weight computation for an unseen label is expressed as:

$$w_{att} = \phi_{avg} \odot w_{avg} + \phi_{att} \odot \left(\frac{1}{k}\sum_{i=1}^{k}\sum_{b=1}^{K_b} Att(\phi_q z_i, k_b)\cdot w_b\right) \tag{2}$$

where *odot* is the Hadamard product, $\phi_{avg}, \phi_{att}, \phi_q$ are learnable parameters, $Att(.,.)$ is an attention kernel, and $\{k_b \in R^d\}_{b=1}^{K_b}$ is a set of K_b learnable keys.

3.3. Multi-level weight-centric (MLWC) representation learning

Fig. 3 provides an overview of our proposed method. The method mainly consists of two techniques: a multi-level feature extractor and a weight-centric feature learning strategy. The former aims to explicitly enforce each single sample feature vector closer to its corresponding classifier weight. Specifically, we construct three levels of feature representations namely mid-level,

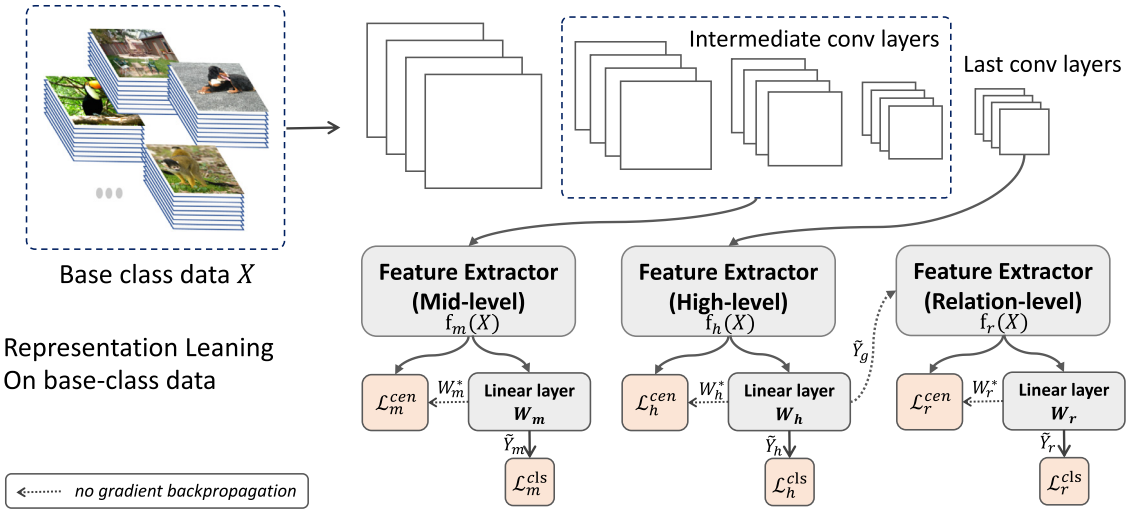

Fig. 3. An overview of our learning framework for representation learning. The framework first construct three levels of feature representations namely mid-level,high-level,and relation-level. For forwarding the networks, the outputs of intermediate layer outputs are detached and fed to the mid-level feature extractor, the output of the last conv layer is forwarded to high-level feature extractor, and the prediction logits of high-level branch are detached and input and sent to the relation-level feature extractor. The three feature extractors are first trained to converge with the classification loss, and then are further fine-tuned with both the classification and the weight-centric loss.

high-level, and relation-level. The mid-level representation captures more subtle discriminative patterns, such as subordinary components of object parts, while the high-level encodes more holistic information. The relation-level is designed to describe the input's category structural relations, like how the input image relates to other categories. The second technique intends to obtain multiple representations that encode different levels of semantic information. Overall, the multi-level extractor improves the representation ability by considering multiple sources of information, and the weight-centric strategy increases the feasibility of generating classifier weights from few-shot sample features. These two techniques can seamlessly join together to provide a simple and effective solution to few-shot learning problems.

3.3.1. Learning weight-centric Feature Embedding

In this subsection, we first review cosine softmax loss for few-shot learning. We then introduce our proposed weight-centric embedding learning strategy. This strategy can be incorporated with cosine softmax loss to facilitate the subsequent step of generating weights from few-shot training examples.

Cosine Softmax Loss. In standard classification framework, Softmax Loss is usually adopted for supervised learning. It generally refers to a Softmax Activation plus a Cross-Entropy Loss. Given an input (x_i, y_i), the softmax loss function is expressed as:

$$\ell_s(x_i, y_i) = -log\left(\frac{exp(w_{y_i}^T f_\Theta(x_i))}{\sum_j exp(w_j^T f_\Theta(x_j))}\right), \tag{3}$$

where $f_\Theta(\cdot)$ is the feature extractor and w_j is the j^{th} column of the weight matrix W of the classifier layer.

However, recent works show the softmax loss fails to learn a feature extractor that generalizes well to unseen categories [21,22]. As discussed previously, the feature extractor of the base model is used to generate weights of novel categories. However, the model transferability gap increases as the distance between tasks grows [24]. Therefore, the more significant difference between the base- and novel tasks, the poorer performance of the few-shot learning model due to the weak transferability of the feature extractor. To ease this issue, Gidaris and Komodakis [21]; Qi et al. [22] propose to adopt cosine softmax loss in learning the base model. Compared with softmax loss, Cosine softmax loss applies l_2-normalization on both the feature vector and the weight vector before the loss calculation, which is expressed as:

$$\tilde{w}_j = \frac{w_j}{\|w_j\|}, \tilde{f}_\Theta(x_i) = \frac{f_\Theta(x_i)}{\|f_\Theta(x_i)\|}. \tag{4}$$

This normalization step will cause the softmax function to fail to produce a one-hot categorical distribution, making the neural networks hard to converge. As suggested in Qi et al. [22], a simple solution to this is to introduce a trainable scale factor s into to the softmax function. Thus, the cosine softmax loss function is expressed as:

$$\ell_{cs}(x_i, y_i; \Theta, W) = -log\left(\frac{s \cdot exp(\tilde{w}_{y_i}^T \tilde{f}_\Theta(x_i))}{\sum_j exp(\tilde{w}_j^T \tilde{f}_\Theta(x_i))}\right). \tag{5}$$

Based on this loss function, Gidaris and Komodakis [21]; Qi et al. [22] learn the feature extractor by minimizing the cost function

$$\mathcal{L}_{cs} = \frac{1}{N}\sum_i^N (\ell_{cs}(x_i, y_i; \Theta, W)) + \lambda R(W), \tag{6}$$

where $\lambda R(W)$ is a weight L_2 regularization term. **Weight-Centric feature learning**. As illustrated in Fig. 4(a) and (b), learning with cosine softmax loss reduces intra-class variations by comparison with original softmax loss. Thus it increases the feasibility to characterize an unseen concept with few-shot examples. Gidaris and Komodakis [21]; Qi et al. [22] assume that the samples of the same class are concentrated in the feature space learned with cosine softmax loss, then the feature embedding of some random samples can be used to approximate the classifier weights. However, this assumption is not strictly held in some cases, such as data with large intraclass variance and small inter-class variance might tend to be scattered in the feature space. To ensure that using one or few embedded points of each category can construct a stable decision boundary, we explicitly constraint a feature point $\tilde{f}(x_i)$ should be near its classifier weight $\tilde{w}_{y_i}$ after the classifier is learned, and the constraint loss is given by

$$\ell_{cen}(x_i, w_{y_i}^*; \Theta) = \| f_\Theta(x_i) - w_{y_i}^* \|^2, \tag{7}$$

where $w_{y_i}^*$ represents the sample x_i's corresponding class weight vector, which specifically refers to the $y_i{}^{th}$ column of a constant matrix W^*. Noted we obtain W^* from the classifier layer after first training the model to converge using the cost function $\mathcal{L}_{cs}$. To couple the constraint with cosine softmax loss, we also apply

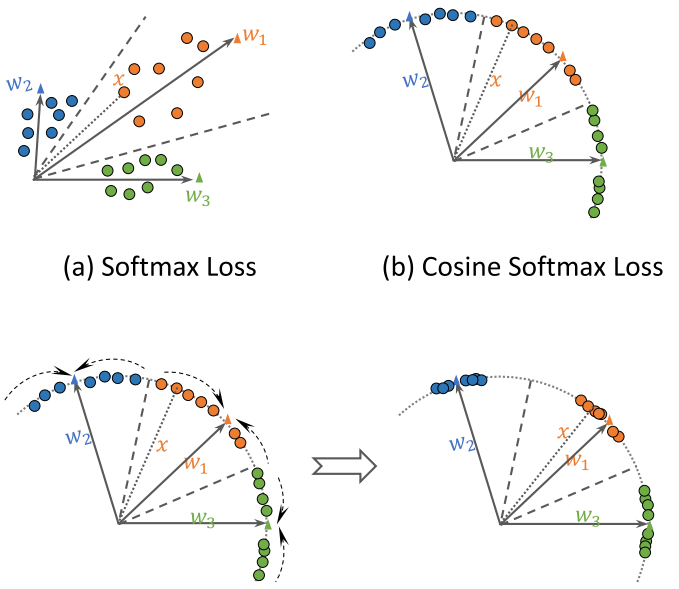

Fig. 4. A geometry interpretation for learning feature space with different loss functions.

l_2-normalization on both the feature vector and the weight vector. Thus, the weight-centric constraint can be rewritten as

$$\ell_{cen}(x_i, w^*_{y_i}; \Theta) = \left\| \frac{f_\Theta(x_i)}{\|f_\Theta(x_i)\|} - \frac{w^*_{y_i}}{\|w^*_{y_i}\|} \right\|^2. \tag{8}$$

By integrating the cosine softmax loss and the weight-centric constraint, we now have the cost function $\mathcal{L}$.

$$\mathcal{L} = \begin{cases} \mathcal{L}_{cs} & \mathcal{L}_{cs} > \epsilon \\ \mathcal{L}_{cen} + \mathcal{L}_{cs}, & otherwise, \end{cases} \tag{9}$$

where $\mathcal{L}_{cen} = \frac{1}{N}\sum_i^N(\ell_{cen}(x_i, w^*_{y_i}))$ and $\mathcal{L}_{cs} > \epsilon$ means that the stopping criteria is not met when training with loss $\mathcal{L}_{cs}$. Since the $\mathcal{L}_{cen}$ required W^* as input, we optimize the cost function using a two-stage algorithm which is detailed in Algorithm 1.

Algorithm 1: Learning weight-centric features.

Input : Base-class Training data $\{X, Y\}$,feature extractor with parameters of Θ
,linear classifier weights $\mathcal{W}$.**Output**: Updated Θ and $\mathcal{W}$
Initialize parameters Θ and $\mathcal{W}$ **while** $\mathcal{L}_{cs}$ *not converge* **do**
  Sample a minibatch of m examples from the training set $\{x^{(1)}, \ldots, x^{(m)}\}$ with corresponding targets $y^{(i)}$;
  Compute gradient: $g_\Theta \longleftarrow \frac{1}{m}\nabla_\Theta \sum_i \mathcal{L}_{cls}(x^{(i)}, y^{(i)}; \Theta, \mathcal{W})$;
  ▷ $\mathcal{L}_{cs}$ `is computed using eq.6`
  Compute gradient: $g_\mathcal{W} \longleftarrow \frac{1}{m}\nabla_\mathcal{W} \sum_i \mathcal{L}_{cls}(x^{(i)}, y^{(i)}; \Theta, \mathcal{W})$;
  update Θ and $\mathcal{W}$;
end
$\mathcal{W}^* \longleftarrow \mathcal{W}$; ▷ `Frozen classifier weights`
while $\mathcal{L}_{centric}$ *and* $\mathcal{L}_{cls}$ *not converge* **do**
  Sample a minibatch of m examples from the training set $\{x^{(1)}, \ldots, x^{(m)}\}$ with corresponding targets $y^{(i)}$;
  Compute gradient:
  $g_\Theta \longleftarrow \frac{1}{m}\nabla_\Theta \sum_i (\mathcal{L}_{cls}(x^{(i)}, y^{(i)}; \Theta, \mathcal{W}^*) + \mathcal{L}_{cen}(x^{(i)}; \Theta, \mathcal{W}^*))$;
  update Θ;
end

As illustrated in Fig. 4(c), the weight-centric constraint push samples closer to their corresponding classifier weights, which brings two advantages. First, it enforces the neural network to learn a feature space with smaller intra-class variance. Moreover, the constraint also implicitly drives samples far away from the decision boundary. This increases the feasibility of constructing a discriminative decision boundary based on a small number of samples.

3.3.2. Multi-level Feature Extractor

A good representation of generalized few-shot learning is it can generalize well to novel concepts while maximizing its original ability to discriminate base categories. A single high-level of feature representation usually has limited capacity to meet these criteria simultaneously. In this subsection, we introduce two additional levels of representation namely mid- and relation-level to complement the representative capacity of high-level representation.

High-level feature extractor is a common practice in most existing few-shot learning methods. As illustrated in Fig. 5(1), It takes inputs from the last convolutional layer and then maps them into an embedding space after applying global-average pooling. This design results in the extracted features naturally capture the global visual discriminative patterns, because of the high-level feature abstraction source and the property of the pooling operation.

Mid-level feature extractor aims to obtain features that focus more on encoding mid-level discriminative patterns. Compared with the high-level features, it exhibits a better generalization ability in representing novel concepts but weaker discriminatory power for the base concepts. This can be attributed to the fact that it tends to abstract information that is less specific to the base concepts. A naive scheme to learn mid-level features is to plug an additional global-extractor head on top of the intermediate layers. However, this solution might still learn features more similar to the high-level ones because of the global average pooling operation, though the input source is switched to the lower layers. To avoid such undesirable effects, we design the mid-level feature extractor, shown in Fig. 5(2). Specifically, we insert a 1x1 Conv layer on top of each intermediate layer and employ global-max pooling to prevent the 1x1 Conv layer from learning global abstraction. Lastly, we concatenate all the intermediate-layer features into one and map it into embedding space to form a compact mid-level representation.

Relation-level feature extractor. As discussed previously, the model's generalization to novel concepts can be improved by incorporating the mid-level representation. However, its ability to classify base classes is degrading when the label space is expanding with more novel classes (some base-class examples might be misclassified to novel classes). Thus, we propose to preserve such ability by encoding more specific information of base classes. Specifically, we introduce another relation-level representation that describes an input using its structural relationships within the base classes. This representation is more specific to base classes than both the high- and mid-level representation. Though it has a poor generalization to novel concepts, it helps strengthen the classification capacity for base classes. As shown in Fig. 5(3), the relation-level extractor takes inputs the predicted logits $\tilde{Y}_h$ from the high-level branch. Here, when the $\tilde{Y}_h$ from a trained model is fed to a softmax layer, the outputs will tend to be a one-hot vector, which fails to describe the data's structural relation over classes. Therefore, we feed $\tilde{Y}_h$ to a softmax function with a high temperature, so that it can encode a richer class structural information of the data. Finally, we use this soften prediction outputs to learn the embedding space that characterizes the similarity of samples according to their categorical distribution.

Jointly Learning multiple feature extractors. As shown in Fig. 3, we learn the three feature extractors using three classification branches that are all based on a single network backbone.

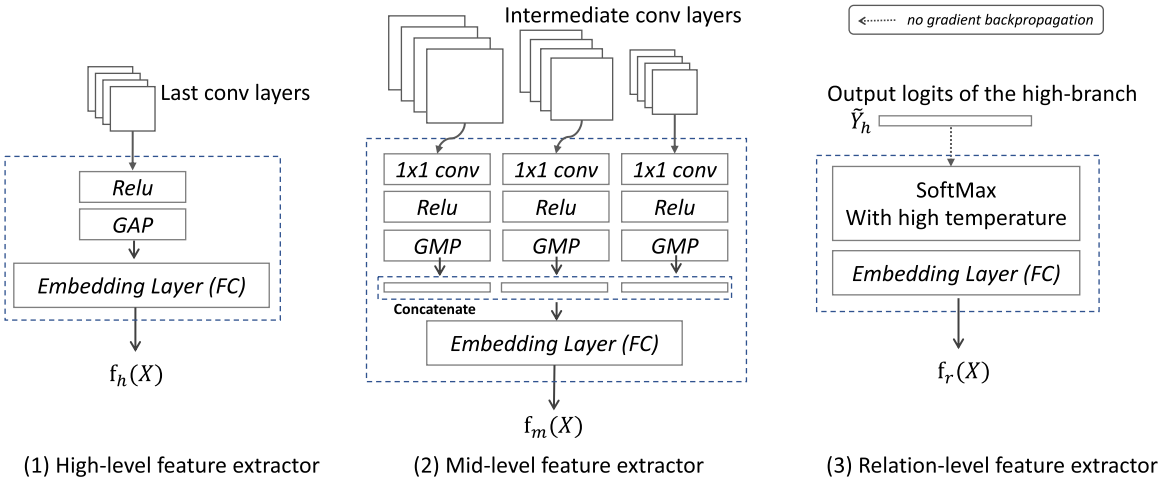

Fig. 5. Sub-network structures for different levels of feature extractor.

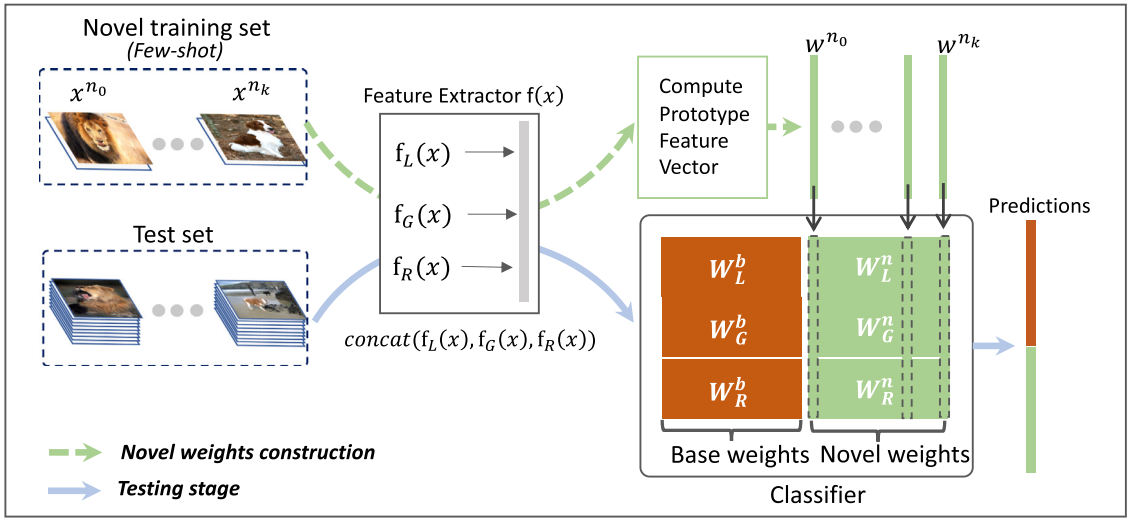

Fig. 6. Utilizing the base models to construct classification models in few-shot learning. We first combine the base modes to obtain a multi-level feature extractor and a base-class weight matrix. Then the feature extractor is used to produce a novel-class weight matrix. Finally, we can construct classification models for few-shot learning by integrating the feature extractor, base- and novel-class weight matrix.

We also apply the weight-centric learning strategy for each branch. Thus, the overall classification loss and weight-centric loss

$$\mathcal{L}_{cs} = \mathcal{L}_{cs}^{m} + \mathcal{L}_{cs}^{h} + \mathcal{L}_{cs}^{r},$$
$$\mathcal{L}_{cen} = \mathcal{L}_{cen}^{m} + \mathcal{L}_{cen}^{h} + \mathcal{L}_{cen}^{r} \tag{10}$$

respectively. Finally, our overall cost function is obtained by substituting these two equations into Eq. (9).

3.4. Few-shot leaning

In the previous section, we describe how our proposed method learns base models on the base-class dataset. In this section, we describe how to utilize these base models to perform few-shot learning. This procedure mainly consists of two operations: model combination and weight generation, which are detailed in the following.

Model combination. After training the base models using our proposed method, we have three base models $M(f_m(x), W_m^b), M(f_h(x), W_h^b)$,and $M(f_r(x), W_r^b)$, which denote the mid-, high-, and relation-level classification model respectively. We simply combine them into a single model $M(f_C(x), W_C^b)$ by concatenating their normalized features and classifier weights separately. Here, $f_C(x) = concat(\frac{f_m(x)}{\|f_m(x)\|}, \frac{f_h(x)}{\|f_h(x)\|}, \frac{f_r(x)}{\|f_r(x)\|})$ forms a multi-level feature extractor and $W_C^b = concat(W_m^b, W_h^b, W_t^b)$ is the classifier weight matrix for base categories. Given a test image x^b, this model can be used to predict the label in the base label space Y^b, that is $argmax(M(f_C(x), W_C^b)) \in Y^b$.

Generating weights for few-shot learning. Now, we can utilize the feature extractor $f_C(\cdot)$ and weight matrix W_C^b to construct different models for different few-shot learning settings. We first construct the weight matrix W_C^n for Y^n using a weight generator (AvgGen [22] or AttGen [21]). Then, we can build classification models $M(f_C(x), W_C^n)$ and $M(f_C(x), [W_C^b, W_C^n])$ for standard and generalized few-shot learning scenario respectively. Here, the weight matrix W_C^n is obtained by stacking each weight vector in order according to its label index in Y^n.

Let Y^b and Y^n denote the base- and novel-label space respectively, we obtain its corresponding weight vector w^y by normalizing the prototype of the given k training samples $\{x_1^y, \ldots, x_k^y\}$.

$$w^y = \frac{\frac{1}{k}\sum_{i=1}^{n} f(x_i^y)}{\|\frac{1}{k}\sum_{i=1}^{n} f(x_i^y)\|}, \tag{11}$$

where $f(.)$ is the multi-level feature extractor derived from the combined base model. Now, Given a unseen label space, we can build classification models $M(f(x), W^n)$ and $M(f(x), [W^b, W^n])$ for

standard and generalized few-shot learning scenario respectively. Here, the weight matrix W^n is obtained by stacking each weight vector in order according to its label index in Y^n, W^b is weight matrix derived from the combined base model.

4. Experiments

4.1. Datasets and evaluation metrics

We validate our proposed method on Low-shot-ImageNet [28] and Low-shot-CUB [22] based on three performance metrics.

Low-shot-ImageNet contains 193 base categories,300 novel categories, 196 base categories, and 311 novel categories respectively. The first two groups are made for validating hyper-parameters, the remaining two groups are used for the final evaluation.

Low-shot-CUB is constructed from Caltech-UCSD bird dataset [49]. The dataset consists of 100 base classes and 100 novel classes. Since each category of this dataset contains only about 30 images, we repeat 20 experiments and take the average top-1 accuracy.

Performance evaluation metrics. Few-shot learning methods are evaluated differently according to different few-shot learning setting. These performance measures mainly differs in the way of constructing a test dataset. To evaluate our proposed method in both standard and generalized setting, we use three evaluation metrics summarized as below:

1) Novel/Novel: the model's performance is measured by the accuracy of novel test examples within the novel label space, that is $D_{test} = \{(x_i, y_i) \in D^n_{test}, y_i \in Y^n\}$.

2) Novel/All: the model's performance is measured only by the accuracy of novel test examples in all label space, that is $D_{test} = \{(x_i, y_i) \in D^n_{test}, y_i \in Y^b \cup Y^n\}$.

3) All: the model's performance is measured only by the accuracy of all test examples in all label space, that is $D_{test} = \{(x_i, y_i) \in D^b_{test} \cup D^n_{test}, y_i \in Y^b \cup Y^n\}$.

Here, standard few-shot learning setting only consider *Novel/Novel* as the major performance measure, while the generalized setting consider results of both *Novel/All* and *All*. We report results of these metrics based on multiple tries. Specifically, in our experiments, we randomly select training images of the novel categories and repeat experiments 100 times, and finally report the mean accuracies within 95% confidence intervals.

4.2. Network architecture and training details

Network architecture. We conduct experiments on the Few-shot-Imagenet benchmark using ResNet-10 and -50 [1] architecture in our learning framework. For experiments on the Few-shot-CUB dataset, as Qi et al. [22] obtained their results based on InceptionV1 [50], we implement our method based on the same network structure for performance comparison.

Training details. For all experiments on imageNet based few-shot benchmarks, we trained our model from scratch for 90 epochs on the base classes. The learning rate starts from 0.1 and is divided by 10 every 30 epochs with a fixed weight decay of 0.0001. We then fine-tune the model for further with the classifier-centric constraint with a small learning rate 0.0001. For the CUB dataset experiment, all the pre-trained models we used are from the Pytorch official model zoo. During the training, the initial learning if 0.001 decreases by 0.1 times at 30 epoch intervals.

4.3. Results and analysis

4.3.1. Low-shot Classification accuracy

We evaluated the performance of the proposed method on two low-shot benchmarks.

Low-shot-ImageNet. Tables 1 and 2 provide the comparative results of different techniques using two network backbones on the large-scale Few-shot-ImageNet dataset. First, we can observe that some existing methods show significant improvement on one evaluation metric but minor on another one. For example, both Weight imprinting [22] and AttGen [21] have better performance than Matching Nets [23] in the "Novel/Novel" setting but similar or even worse results in the "Novel/ALL" setting. In comparison, our approach consistently achieves the best results on all evaluation metrics. Specifically, using the same weight generator AttGen, our method significantly outperforms the current best model TRAML [51] in testing both novel-class and all-class classification accuracy. Besides, without learning the weight generator, our proposed method also achieves a comparable performance to the current top-performing methods that require training a weight generator. For instance, compared to the TRAML method that needs to learn an attention-based weight generator, our approach obtains a similar performance using the mean feature as classifier weights. All these results indicate that our learned representation yields a better generalization ability and versatility for FSL learning.

Low-shot-CUB. Since existing method reported on this dataset is based on Inception V1 network, we first evaluate our method with the same backbone network. Table 3 shows performance comparison result of different approaches. Our proposed method outperforms all the comparing method by a large margin in all evaluation metrics. For instance, our method achieves top-1 accuracies of 30.72% and 37.65% under the 1 and 2 shot settings respectively, the previous best results are 21.40% and 28.69%. To evaluate out method's effectiveness on this dataset when using different network architecture, we further use the Resnet-50 as backbone for both the Imprinting and our method and compare their performance. Table 4 shows the corresponding results and, again, demonstrates our method's superior performance in low-shot learning.

Cross-domain performance of low-shot learning. We investigate the transferability of different levels of representations in the FSL setting. To achieve this, we perform a cross-domain evaluation, where we evaluate the learned model on both the same-domain and different domain data. Specifically, we first train a model on the base-class data from the ImageNet dataset. Then we evaluate it on both the ImageNet and the Caltech-UCSD bird dataset [49]. Table 5 presents the comparison results obtained based on the Resnet-50 backbone and the Avg weight generator. First, we can observe that learning with weight-centric constraint improves performance on both the same-domain and cross-domain settings. Also, the mid-level features achieve the best accuracy in cross-domain testing while the relation-level performs the worst. This result reveals that the mid-level representation exhibit strong transferability in the FSL setting. Furthermore, the proposed multi-level representation achieves the best accuracy on the same-domain data and obtains comparable performance with the mid-level features. This indicates that using multi-level features for FSL help improve generalization ability and handle domain shift problem.

4.3.2. Analysis and ablation study

Effectiveness of the classifier-centric constraint. To verify the effectiveness of the classifier-centric constraint, we established the following experiments. First, we train two ConvNet models on the base class data, with and without classifier-centric constraints to learn the two feature spaces. Then we randomly sample some samples from each class of the base class dataset to construct two classifiers to classify the test set. Finally, by evaluating their classification performance, it is indicated in which feature space the sample can construct a better decision boundary. The experimental results are shown in Table 6. We can observe that the

Table 1
Comparison of top-5 accuracy with the state-of-art methods using Resnet-10 on the Low-shot-ImageNet dataset. Best are bolded. * indicates that we get 5 random crops from each training example, then use the average feature as the weight of novel class.

	Novel / Novel					Novel / All					All				
Method	n = 1	2	5	10	20	n = 1	2	5	10	20	n = 1	2	5	10	20
Pro. Nets [39] (from [47])	39.4	54.4	66.3	71.2	73.9	-	-	-	-	-	49.5	61.0	69.7	72.9	74.6
Log. Reg. (from [10])	38.4	51.1	64.8	71.6	76.6	-	-	-	-	-	40.8	49.9	64.2	71.9	76.9
Log. Reg w/G. (from [10])	40.7	50.8	62.0	69.3	76.5	-	-	-	-	-	52.2	59.4	67.6	72.8	76.9
Pro. Mat. Nets [10]	43.3	55.7	68.4	74.0	77.0	-	-	-	-	-	55.8	63.1	71.1	75.0	77.1
Pro. Mat. Nets w/G [10]	45.8	57.8	69.0	74.3	77.4	-	-	-	-	-	57.6	64.7	71.9	75.2	77.5
SGM w/G [28].	-	-	-	-	-	32.8	46.4	61.7	69.7	73.8	54.3	62.1	71.3	75.8	78.1
Batch SGM [28]	-	-	-	-	-	23.0	42.4	61.9	69.9	74.5	49.3	60.5	71.4	75.8	78.5
Mat. Nets [23] (from [10,28])	43.6	54.0	66.0	72.5	76.9	41.3	51.3	62.1	67.8	71.8	54.4	61.0	69.0	73.7	76.5
Wei. Imprint* + AvgGen [22]	44.05	55.42	68.06	73.96	77.21	38.70	51.36	65.89	72.60	76.21	56.73	63.66	71.04	74.05	75.47
	±.21	±.16	±.09	±.07	±.05	±.21	±.17	±.09	±.07	±.05	±.13	±.10	±.06	±.04	±.03
AvgGen (with retraining) [21]	45.23	56.90	68.68	74.36	77.69	39.33	50.27	63.16	69.56	73.47	54.65	64.69	72.35	76.18	78.46
	±.25	±.16	±.09	±.06	±.06	±.25	±.16	±.11	±.07	±.06	±.15	±.10	±.06	±.04	±.04
AttGen [21]	46.02	57.51	69.16	74.84	78.81	40.79	51.51	63.77	70.07	74.02	58.16	65.21	72.72	76.65	78.74
	±.25	±.15	±.09	±.06	±.05	±.25	±.15	±.12	±.07	±.06	±.15	±.09	±.06	±.04	±.03
TRAML [51] + AttGen	48.1	59.2	70.3	76.4	79.4	-	-	-	-	-	59.2	66.2	73.6	77.3	80.2
MLWC + AvgGen	48.22	58.77	69.71	74.45	76.91	44.06	55.83	68.15	73.36	76.07	58.96	65.18	71.28	73.63	74.78
	±.12	±.09	±.05	±.03	±.02	±.12	±.09	±.05	±.04	±.02	±.07	±.05	±.03	±.02	±.02
MLWC* + AvgGen	49.09	59.66	70.26	74.72	77.04	45.56	57.12	68.85	73.73	76.24	59.37	65.48	71.36	73.63	74.72
	±.11	±.08	±.04	±.03	±.02	±.11	±.09	±.05	±.03	±.02	±.07	±.05	±.03	±.02	±.02
MLWC* + AttGen	**50.87**	**62.13**	**72.61**	**77.02**	**79.67**	**46.18**	**57.21**	**68.63**	**73.64**	**76.59**	**61.72**	**68.58**	**75.35**	**78.29**	**80.03**
	±.22	±.15	±.09	±.06	±.23	±.15	±.09	±.09	±.07	±.05	±.14	±.08	±.06	±.05	±.03

Table 2
Comparison of top-5 accuracy with the state-of-art methods using Resnet-50 on the Low-shot-ImageNet dataset.Best are bolded. * indicates that we get 5 random crops from each training example, then use the average feature as the weight of novel class.

	Novel / Novel					Novel / All					All				
Method	n = 1	2	5	10	20	n = 1	2	5	10	20	n = 1	2	5	10	20
Mat. Nets [23] (from [10])	53.5	63.5	72.7	77.4	81.2	-	-	-	-	-	64.9	71.0	77.0	80.2	82.7
Pro. Nets [39]	49.6	64.0	74.4	78.1	80.0	-	-	-	-	-	61.4	71.4	78.0	80.0	81.1
Pro. Mat. Nets w/G [10]	54.7	66.8	77.4	81.4	83.8	-	-	-	-	-	65.7	73.5	80.2	82.8	84.5
SGM w/G. (from [10])	-	-	-	-	-	45.1	58.8	72.7	79.1	82.6	63.6	71.5	80.0	**83.3**	**85.2**
MLWC + AvgGen	57.12	68.28	77.77	81.80	83.72	53.48	65.05	76.59	80.95	83.07	67.49	73.36	79.87	81.98	82.95
	±.20	±.14	±.07	±.07	±.04	±.23	±.13	±.08	±.08	±.04	±.14	±.08	±.05	±.05	±.02
MLWC*+AvgGen	**57.97**	**69.08**	**78.19**	**81.99**	**83.80**	**54.82**	**66.93**	**77.12**	**81.22**	**83.16**	**68.01**	**74.72**	**79.98**	81.99	82.88
	±.20	±.15	±.06	±.07	±.03	±.22	±.05	±.05	±.08	±.03	±.13	±.09	±.05	±.05	±.02

Table 3
Comparison of top-1 accuracy with the state-of-art methods on the Few-shot-Cub dataset. * indicates that we get 5 random crops from each training example, then use the average feature as the weight of novel class.

	Novel / Novel					Novel / All					All				
Method	n = 1	2	5	10	20	n = 1	2	5	10	20	n = 1	2	5	10	20
Gen. + Cla [28] (from [22])	-	-	-	-	-	18.56	19.07	20.00	20.27	20.88	45.42	46.56	47.79	47.88	48.22
Mat. Nets [23] (from [22])	-	-	-	-	-	13.45	14.75	16.65	18.18	25.77	41.71	43.15	44.46	45.65	48.63
Imprinting [22]	-	-	-	-	-	21.26	28.69	39.52	45.77	49.32	44.75	48.21	52.95	55.99	57.47
Imprinting* [22]	-	-	-	-	-	21.40	30.03	39.35	46.35	49.80	44.60	48.48	52.78	56.51	57.84
MLWC	32.35	39.78	49.47	54.67	57.37	30.72	37.65	48.17	53.56	56.45	49.80	53.41	57.87	**60.46**	**61.61**
MLWC*	**33.56**	**40.82**	**50.28**	**54.67**	**57.53**	**30.87**	**39.01**	**49.17**	**53.66**	**56.61**	**49.96**	**53.73**	**58.18**	60.30	61.60

Table 4
Comparison of top-1 accuracy with the state-of-art methods on the Few-shot-Cub dataset. * indicates that we get 5 random crops from each training example, then use the average feature as the weight of novel class.

	Novel / Novel					Novel / All					All				
Method	n = 1	2	5	10	20	n = 1	2	5	10	20	n = 1	2	5	10	20
Imprinting* [22] (Resnet50*)	32.15	40.48	52.41	57.93	61.72	26.24	35.79	49.31	55.31	59.38	52.43	56.83	62.89	65.53	67.27
MLWC	35.91	44.91	56.95	62.48	66.01	33.54	43.47	56.21	61.96	65.61	55.45	59.58	64.94	67.32	68.78
MLWC*	**36.96**	**45.53**	**57.43**	**63.03**	**66.35**	**34.91**	**44.21**	**56.81**	**62.52**	**65.96**	**55.60**	**59.66**	**65.02**	**67.46**	**68.89**

feature space learned with cosine softmax loss achieve poor accuracy, that indicates the sample points in this space might be scattered and not close to the classifier weight. By applying the classifier-centric constraint, the accuracy is significantly improved. This demonstrates that the feature space learned with classifier-centric constraint is more suitable for building classifiers using samples. We further evaluate the classifier-centric constraint under different evaluation metrics and provide the results in Fig. 7. We can see that our proposed constraint improves the baseline consistently in three evaluation metrics. More importantly, the improvements under the "ALL/ALL" setting are the most significant, revealing that the classifier-centric constraint exhibits superiority

Table 5
The performance of using different levels of representation for few-shot learning on the same task (Generic object classification) and another different task (Fine-grained object classification). Top-5 accuracy of the novel categories in the novel label space (Novel/Novel) is reported. WC denotes our proposed weight-centric constrain. Best are bolded.

	Novel classes from ImageNet					Novel classes from CUB2011				
Method	n = 1	2	5	10	20	n = 1	2	5	10	20
High-level (baseline)	51.56	63.67	74.78	79.68	82.45	30.55	40.76	53.68	60.79	65.54
High-level (baseline)+WC	54.24	65.71	75.75	81.33	82.80	35.92	47.67	61.92	69.35	73.42
Mid-level	51.59	63.80	75.57	80.60	83.21	35.99	**48.40**	**62.51**	**70.26**	**74.92**
Relation-level	48.94	58.64	69.23	73.32	75.65	24.45	32.19	40.92	46.18	49.18
Multi-level	**55.50**	**67.51**	**78.26**	**82.75**	**85.00**	**36.15**	48.34	62.44	69.94	74.37

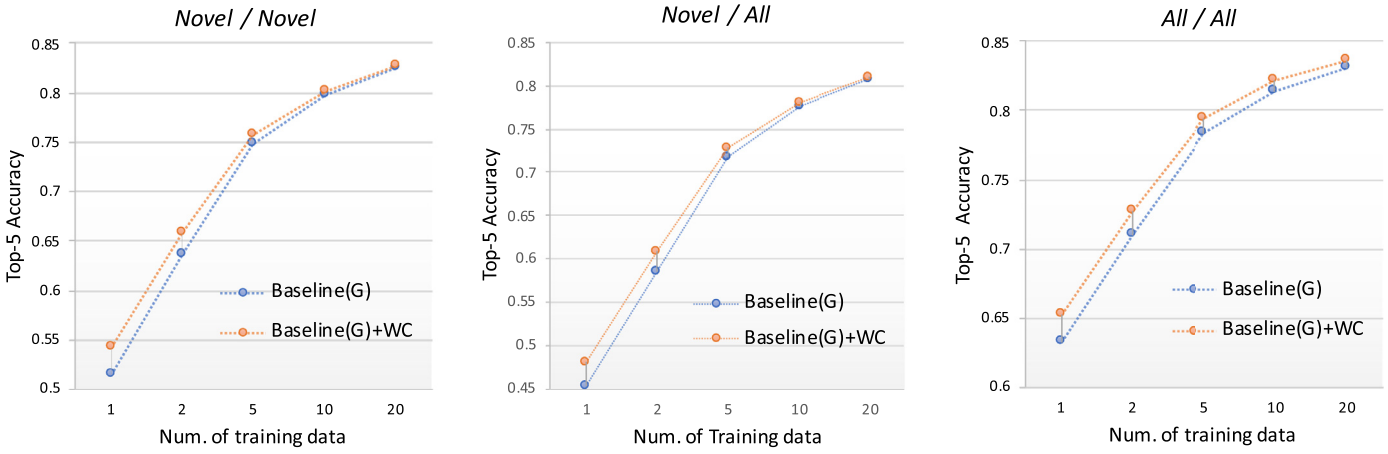

Fig. 7. Top-1 Classification accuracy of few-shot setting on CUB set. Here, baseline refers to the feature space learned with cosine softmax loss, WC denotes our proposed weight-centric constrain.

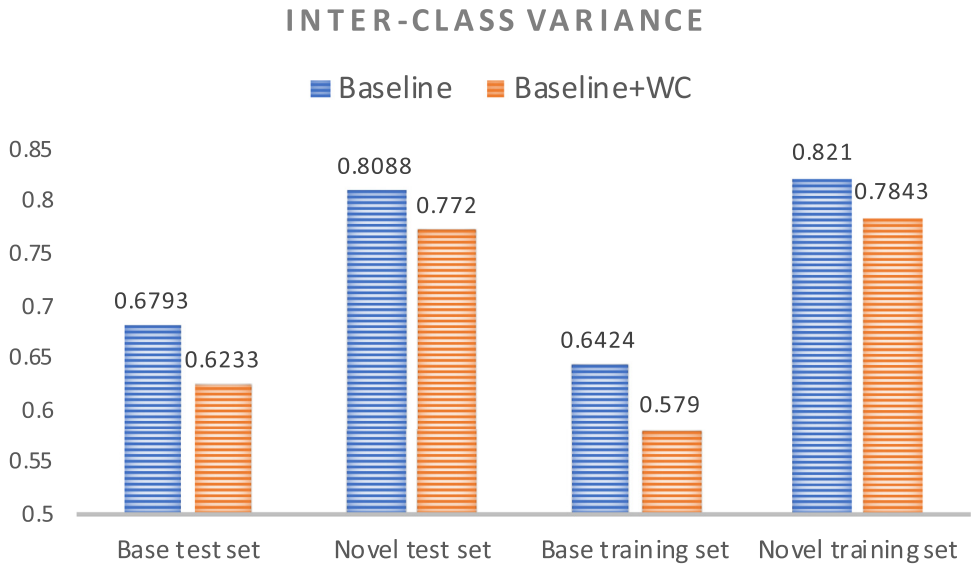

Fig. 8. Comparison of the intra-class variance between two feature spaces both learned on base training set. Here, baseline refers to the feature space learned with cosine softmax loss, WC denotes our proposed weight-centric constrain. Noted that we report the average intra-class variance for each dataset.

Table 6
Top-1 Classification accuracy on CUB Base-class test set using samples as the classifier in two feature spaces.

Method	n=1	2	5	10	20	Classifier
Baseline [21,22]	53.96	62.88	69.55	71.56	73.42	81.80
Baseline + WC	**69.93**	**74.94**	**78.30**	**78.99**	**79.68**	81.71

in generalized few-shot learning. Fig. 8 shows a comparison of the intra-class variance between two feature spaces learned by with and without weight-centric constraint. It can be seen that training with weight-constraint reduce intra-class variance on both training and test set, and also both base and novel class data.

The contribution of each component. We conduct ablation study to compare the performance of different levels representation in the FSL setting. Table 7 provides an ablation study on the Few-shot-imagenet benchmarks to observe the effect of each element. On the one hand, we can see that when evaluating only the novel label space, adding the weight-centric and mid-level component in sequence continuously improves the performance. This demonstrates that both pieces help enhance the model generalization ability, which also implies that increasing the prototype-ability

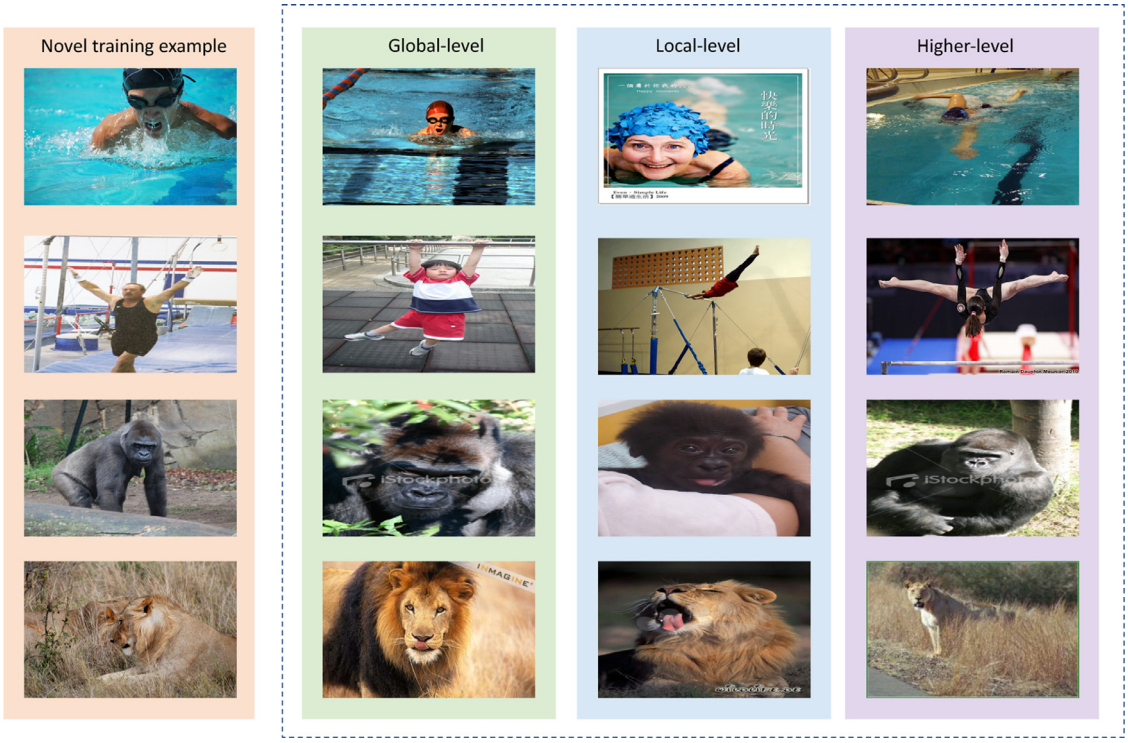

Fig. 9. Some successful exemplars using our proposed method. The first column shows a single training image of novel class, all images in the remaining three columns are correctly predicted by using the proposed multi-level representation. The second column shows some successful predictions using only global-level features but they are mis-classified if using local or higher-level representation, and so on for the second and the third column.

Table 7
Oblation study experiments on the ImageNet based few-shot benchmark. *H, M*,and *R* refer to High-, Mid-, and relation-level features, respectively. *WC* refers to using weight-centric learning strategy.

	Novel / Novel			Novel / All		
	n=1	2	5	n=1	2	5
H(baseline)	51.56	63.67	74.78	45.26	58.53	71.80
H+WC	54.24	65.71	75.75	47.95	60.77	72.91
(H+WC)+M	56.96	68.50	78.58	50.79	64.30	76.52
(H+WC+M)+R	57.12	68.28	77.77	53.48	65.82	76.95

and transferability of feature representation can benefit few-shot learning. On the other hand, incorporating relation-level features does not further raise the performance in this setting. However, it shows a significant improvement under the "novel/all" evaluation metric. This indicates that the relation-level features have weaker generalization to novel classes but can effectively prevent novel-class data from being classified into the base categories.

We also provide some prediction results in Fig. 9, which can be used to intuitively analyze the few-shot learning ability of different representation. For example, the test images in the second column mostly contain some patterns (e.g., objects or parts of objects) which are very similar to those occurs in the training examples, while the similarities between images in the last two columns and the training images tend to be subtle.

5. Conclusion

This work investigates the problem of feature representation in few-shot learning. To improve the representation power for unseen categories and weight generation capacity in feature learning, we proposed a multi-level weight-centric representation learning approach. The method first incorporated mid- and relation-level features with high-level to enhance representation capacity. Also, a classifier-centric learning strategy was proposed to allow a few sample features to construct a more discriminative classifier. Compared with existing methods, the method increases the feasibility of building a discriminative decision boundary based on a few samples. Also, it improves the transferability for characterizing novel classes and preserve classification capability for base classes. In experiments, we extensively evaluate our approach on two low-shot classification benchmarks and demonstrate its effectiveness in improving generalization. Our proposed method can also benefit other tasks such as zero-shot learning and image retrieval, in which feature extractors play a critical role. However, one drawback of our approach is that it constructed multi-level features by concatenating multiple features, introducing redundancy in learning. Therefore, we will investigate how to learn a compact representation from numerous information sources in future work. In addition, our proposed method may suffer from forgetting the base-class knowledge when more novel classes are expanded into the classification model. Thus, our future work will investigate how to prevent forgetting issues in long-term incremental learning settings.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[2] S. Huang, X. Wang, D. Tao, Stochastic partial swap: Enhanced model generalization and interpretability for fine-grained recognition, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 620–629.
[3] L. Zhang, S. Huang, W. Liu, Learning sequentially diversified representations for fine-grained categorization, Pattern Recognit 121 (2022) 108219.

[4] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, A. Oliva, Learning deep features for scene recognition using places database, Advances in Neural Information Processing Systems, 2014.
[5] L. Hu, S. Huang, S. Wang, W. Liu, J. Ning, Do We Really Need Frame-by-Frame Annotation Datasets for Object Tracking?, Association for Computing Machinery, New York, NY, USA, 2021, p. 49494957.
[6] L. Zhang, S. Huang, W. Liu, Enhancing mixture-of-experts by leveraging attention for fine-grained recognition, IEEE Trans Multimedia (2021).
[7] K. Lee, S. Maji, A. Ravichandran, S. Soatto, Meta-learning with differentiable convex optimization, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 10657–10665.
[8] Y. Lifchitz, Y. Avrithis, S. Picard, A. Bursuc, Dense classification and implanting for few-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9258–9267.
[9] B. Oreshkin, P.R. López, A. Lacoste, Tadam: Task dependent adaptive metric for improved few-shot learning, Advances in Neural Information Processing Systems, 2018.
[10] Y.-X. Wang, R. Girshick, M. Hebert, B. Hariharan, Low-shot learning from imaginary data, Proceedings of the IEEE conference on computer vision and pattern recognition (2018).
[11] L. Zhang, X. Chang, J. Liu, M. Luo, M. Prakash, A.G. Hauptmann, Few-shot activity recognition with cross-modal memory network, Pattern Recognit 108 (2020) 107348.
[12] L. Fei, B. Zhang, J. Wen, S. Teng, S. Li, D. Zhang, Jointly learning compact multi-view hash codes for few-shot FKP recognition, Pattern Recognit 115 (2021) 107894.
[13] R. Singh, V. Bharti, V. Purohit, A. Kumar, A.K. Singh, S.K. Singh, Metamed: few-shot medical image classification using gradient-based meta-learning, Pattern Recognit (2021) 108111.
[14] G. Kim, H.-G. Jung, S.-W. Lee, Spatial reasoning for few-shot object detection, Pattern Recognit 120 (2021) 108118.
[15] J. Xu, Q. Du, Learning transferable features in meta-learning for few-shot text classification, Pattern Recognit Lett 135 (2020) 271–278.
[16] Y. Wang, L. Zhang, Y. Yao, Y. Fu, How to trust unlabeled data instance credibility inference for few-shot learning, IEEE Trans Pattern Anal Mach Intell (2021).
[17] T. Chen, L. Lin, X. Hui, R. Chen, H. Wu, Knowledge-guided multi-label few-shot learning for general image recognition, IEEE Trans Pattern Anal Mach Intell (2020).
[18] X. Chen, L. Yao, T. Zhou, J. Dong, Y. Zhang, Momentum contrastive learning for few-shot COVID-19 diagnosis from chest CT images, Pattern Recognit 113 (2021) 107826.
[19] Y. Song, C. Chen, Mppcanet: a feedforward learning strategy for few-shot image classification, Pattern Recognit 113 (2021) 107792.
[20] W. Zhu, W. Li, H. Liao, J. Luo, Temperature network for few-shot learning with distribution-aware large-margin metric, Pattern Recognit 112 (2021) 107797.
[21] S. Gidaris, N. Komodakis, Dynamic few-shot visual learning without forgetting, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018.
[22] H. Qi, M. Brown, D.G. Lowe, Low-shot learning with imprinted weights, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018.
[23] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al., Matching networks for one shot learning, Advances in Neural Information Processing Systems, 2016.
[24] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks? Advances in Neural Information Processing Systems, 2014.
[25] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, Web-scale training for face identification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015.
[26] L. Bertinetto, J.F. Henriques, P. Torr, A. Vedaldi, Meta-learning with differentiable closed-form solvers, in: International Conference on Learning Representations (ICLR), 2018.
[27] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: International Conference on Machine Learning, 2017.
[28] B. Hariharan, R.B. Girshick, Low-shot visual recognition by shrinking and hallucinating features, in: IEEE International Conference on Computer Vision, 2017.
[29] S. Ravi, H. Larochelle, Optimization as a model for few-shot learning, in: International Conference on Learning Representations (ICLR), 2017.
[30] C. Xing, N. Rostamzadeh, B. Oreshkin, P.O.O. Pinheiro, Adaptive cross-modal few-shot learning, in: Advances in Neural Information Processing Systems, 2019, pp. 4847–4857.
[31] S.W. Yoon, J. Seo, J. Moon, Tapnet: Neural network augmented with task-adaptive projection for few-shot learning, in: International Conference on Machine Learning, 2019, pp. 7115–7123.
[32] G. Koch, R. Zemel, R. Salakhutdinov, Siamese neural networks for one-shot image recognition, in: International Conference on Machine Learning Deep Learning Workshop, 2015.
[33] A. Li, T. Luo, T. Xiang, W. Huang, L. Wang, Few-shot learning with global class representations, in: IEEE international conference on computer vision, 2019, pp. 9715–9724.
[34] Y. Liu, J. Lee, M. Park, S. Kim, Y. Yang, Transductive propagation network for few-shot learning, International Conference on Learning Representations (ICLR) (2019).
[35] S. Qiao, C. Liu, W. Shen, A.L. Yuille, Few-shot Image Recognition by Predicting Parameters from Activations, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018.
[36] A.A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, R. Hadsell, Meta-learning with latent embedding optimization, International Conference on Learning Representations (ICLR) (2019).
[37] Q. Sun, Y. Liu, T.-S. Chua, B. Schiele, Meta-transfer learning for few-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 403–412.
[38] L. Fei-Fei, R. Fergus, P. Perona, One-shot learning of object categories, TPAMI 28 (4) (2006) 594–611.
[39] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, Advances in Neural Information Processing Systems, 2017.
[40] F.S.Y. Yang, L. Zhang, T. Xiang, P.H.S. Torr, T.M. Hospedales, Learning to compare: relation network for few-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018.
[41] Z. Ji, X. Chai, Y. Yu, Z. Zhang, Reweighting and information-guidance networks for few-shot learning, Neurocomputing 423 (2021) 13–23.
[42] Y. Guo, N.-M. Cheung, Attentive weights generation for few shot learning via information maximization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 13499–13508.
[43] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C.F. Wang, J.-B. Huang, A closer look at few-shot classification, in: International Conference on Learning Representations (ICLR), 2019.
[44] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: A deep convolutional activation feature for generic visual recognition, in: International Conference on Machine Learning, 2014.
[45] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, Overfeat: Integrated recognition, localization and detection using convolutional networks, 2014.
[46] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European Conference on Computer Vision, 2014.
[47] P. Agrawal, R. Girshick, J. Malik, Analyzing the performance of multilayer neural networks for object recognition, in: European Conference on Computer Vision, 2014.
[48] H. Azizpour, A. Sharif Razavian, J. Sullivan, A. Maki, S. Carlsson, From generic to specific deep representations for visual recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015.
[49] C. Wah, S. Branson, P. Welinder, P. Perona, S. Belongie, The caltech-ucsd birds-200-2011 dataset (2011).
[50] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015.
[51] A. Li, W. Huang, X. Lan, J. Feng, Z. Li, L. Wang, Boosting few-shot learning with adaptive margin loss, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 12576–12584.

Mingjiang Liang is currently pursuing a Ph.D. from the University of Technology Sydney. She received her master's degree from Northwestern Polytechnical University. Her current research interest mainly focuses on few-shot learning.

Shaoli Huang received his Ph.D. degree from the University of Technology Sydney in 2017. He is currently a senior researcher at Tencent AI lab. He previously worked as a research fellow in UBTECH Sydney AI Centre at The University of Sydney. His research interests include deep learning, pose estimation, object detection, and fine-grained object recognition.

Shirui Pan is a Senior Lecturer with the Department of Data Science & AI, Faculty of Information Technology, Monash University. Prior to this, he was a Lecturer with the Centre for Artificial Intelligence (CAI), School of Software, Faculty of Engineering and Information Technology, University of Technology Sydney (UTS). Shirui received his Ph.D. degree in computer science from UTS, Australia. His research interests include data mining and machine learning, specializing in graph mining and network analysis.

Mingming Gong is a lecturer in Data Science at the School of Mathematics and Statistics, University of Melbourne (UoM). He is part of the Melbourne Deep Learning Group. His research interests lie in machine learning, artificial intelligence, and data science, especially in causal reasoning, transfer learning, and deep learning. He develops computational methods for causal reasoning from observational data and exploits causal knowledge to build more intelligent machine learning algorithms.

Wei Liu is an Associate Professor in Machine Learning, and the Director of Future Intelligence Research Lab, in the School of Computer Science, the University of Technology Sydney (UTS). He obtained his Ph.D. degree in Machine Learning research at the University of Sydney (USyd). His current research focuses are adversarial machine learning, cybersecurity, game theory, multimodal machine learning, natural language processing, and intrusion detection.