# Learning Intact Features by Erasing-Inpainting for Few-shot Classification

**Junjie Li, Zilei Wang*, Xiaoming Hu**
Department of Automation, University of Science and Technology of China
hnljj@mail.ustc.edu.cn, zlwang@ustc.edu.cn, cjdc@mail.ustc.edu.cn

## Abstract

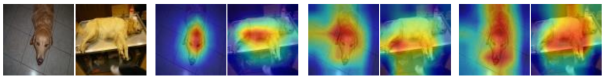
Few-shot classification aims to categorize the samples from unseen classes with only few labeled samples. To address such a challenge, many methods exploit a base set consisting of massive labeled samples to learn an instance embedding function, *i.e.*, image feature extractor, and it is expected to possess good transferability among different tasks. Such characteristics of few-shot learning are essentially different from that of traditional image classification only pursuing to get discriminative image representations. In this paper, we propose to learn intact features by erasing-inpainting for few-shot classification. Specifically, we argue that extracting intact features of target objects is more transferable, and then propose a novel cross-set erasing-inpainting (CSEI) method. CSEI processes the images in the support set using erasing and inpainting, and then uses them to augment the query set of the same task. Consequently, the feature embedding produced by our proposed method can contain more complete information of target objects. In addition, we propose task-specific feature modulation to make the features adaptive to the current task. The extensive experiments on two widely used benchmarks well demonstrates the effectiveness of our proposed method, which can consistently get considerable performance gains for different baseline methods.

## Introduction

Image classification with massive labeled data has achieved great success benefiting from the power of deep convolutional neural networks (DCNN) (He et al. 2016; Krizhevsky, Sutskever, and Hinton 2012; Szegedy et al. 2015, 2016; Hu, Shen, and Sun 2018). But it is still very challenging for DCNN to quickly learn a new concept from few samples. Few-shot classification exactly aims to categorize the samples from the classes with only few labeled samples. Conventionally, there are two fundamental concepts in few-shot classification: base set (seen class) and novel set (unseen class). Particularly, each class in the base set has a lot of labeled samples, while the class in the novel set only contains few samples. Note that there is no class overlap between the base and novel sets. For few-shot learning, we need train a classifier on the base set, and then use it to category the query samples (*i.e.*, query set) from the novel set into the unseen classes having few labeled samples (*i.e.*, support set).

Currently, the metric-based methods can achieve promising performance for few-shot learning (Ye et al. 2020; Li et al. 2019a; Vinyals et al. 2016; Sung et al. 2018). These methods first adopt a feature extractor (embedding function) to translate samples into the feature embedding, and then employ a metric function to calculate the similarity between a test sample in the query set and each class represented by few samples in the support set. Finally, the test sample is classified according to the nearest-neighbor rule. Evidently, how to design the feature extractor and metric function are the two key problem of these methods, which critically determine the classification performance.

On the other hand, the episode training method (Vinyals et al. 2016) is proposed to train the models. It samples the tasks from the base set by imitating the settings in test stage, which are called *episodes*, and then trains the model using the sampled tasks. Through maintaining the consistency of the sampling task over the seen and unseen classes, it is expected the model trained on the base set can well transfer to the novel set. With the help of episode training, many methods dedicate to building good metric functions, and a simple CNN (*e.g.*, ResNet-12) is usually trained for feature extraction. For examples, DSN (Simon et al. 2020) proposes to exploit the subspace distance to define a classifier which is robust against perturbations. EMD (Zhang et al. 2020b) employs Earth Mover's Distance to calculate the similarity of local features between the query sample and support sample.



(a) (b) (c) (d)

Figure 1: Activation maps for different training methods: (a) input, (b) traditional classification model, (c) CAN (baseline), and (d) CSEI (ours). Here warmer color indicates higher values. Best viewed in color.

Actually, the requirement of few-shot classification to feature extractor is different from the traditional image classification. In the traditional image classification, there are lots of labeled samples for training and thus the learned model

*Corresponding Author

can nearly represent the real distribution of test samples. Consequently, the classifier can usually determine the category of an image by only capturing the discriminative information from partial area of target objects (Wei et al. 2017), as shown in Figure 1(b). However, such a way of feature extraction may not work well for few-shot learning, which uses the model trained on the base set to perform classification for unseen classes by matching the query sample and support samples. To be specific, the samples in the support set and query set may respond to different parts of target objects, and Figure 1(c) shows a case. When a dog faces the camera, the dog's head is highlighted due to its discrimination. But when only a small part of the head appears in the image, the main response area is migrated to other parts such as legs. If such two situations occur corresponding to the support set and query set, it would be very difficult to achieve a good matching since the head and leg are greatly different. Therefore, the feature extractor for few-shot learning is required to have good transferability in instance matching other than capturing the discriminative information.

In this paper, we propose to enhance the transferability of the embedding function by explicitly enforcing it to extract intact features of target objects. In particular, we propose a novel cross-set erasing-inpainting (CSEI) method. Specifically, during the training, CSEI first erases the most discriminative area of the images in the support set and then completes the erased images using image inpainting. After that, the processed images are added into the query set of the same task. Through such cross-set data augment, the produced feature embedding can represent more area of target objects, as shown in Figure 1(d). In addition, we propose a task-specific feature modulation (TSFM) module to make the features adaptive to the current task, which consists of the task-level channel-wise attention and instance-level spatial attention. We experimentally evaluate the proposed method on two benchmark datasets, namely, miniImageNet (Vinyals et al. 2016) and tieredImageNet (Ren et al. 2018) and the results demonstrate the effectiveness of our proposed method.

The main contributions of this work are summarized as follows: (1) We propose to enhance the transferability of the embedding function in instance matching other than extracting the discriminative information for few-shot learning. (2) We propose a novel *cross-set erasing-inpainting* for network training, which can enforce the embedding function to extract more information from different regions of target objects. (3) We integrate the proposed techniques into the representative few-shot learning methods, and experimentally evaluate them on the 5-way 1-shot and 5-way 5-shot settings. It is shown that our proposed method can bring the performance gain consistently.

## Related Work

### Few-shot Learning

Here we divide the few-short learning methods into two broad categories according to the adopted techniques. Particularly, this work follows the metric-based approaches with aims of getting better feature embedding of images.

**Optimization-based approaches** For these approaches (Finn, Abbeel, and Levine 2017; Munkhdalai et al. 2018; Antoniou, Edwards, and Storkey 2018; Li et al. 2019c; Sun et al. 2019), the model is optimized in a few steps to quickly adapt to the current task (episode). For example, MAML (Finn, Abbeel, and Levine 2017) targets to find a single set of initialization parameters, on which the model can be adapted to new tasks with a few fine-tuning. LEO (Rusu et al. 2019) proposes to find the most suitable parameter for the current task by optimizing a latent embedding in a low-dimensional space.

**Metric-based approaches** This type of approaches (Hou et al. 2019; Ye et al. 2020; Zhang et al. 2020b; Li et al. 2019a; Simon et al. 2020) first learn a feature embedding of images and then employ a distance function to determine the category of test samples. Most of existing methods focus on how to design a good distance metric, where an off-the-shelf CNN is usually adopted to produce the image features. For example, DN4 (Li et al. 2019b) adopts a local descriptor based measure instead of the image-level feature based measure to calculate the similarity between the support feature and the query feature. CAN (Hou et al. 2019) proposes a Cross Attention Module to find the semantic relevance between the query image and support images to more precisely calculate the similarity. EMD (Zhang et al. 2020b) employs the Earth Mover's Distance as a metric to obtain more discriminative information with local features. However, when the discriminative clues extracted from the support samples are not significant or do not exist in the query samples, it is difficult to match them correctly even using these advanced distance metrics. In addition, some methods consider the relevance between features and the current task, and then proposed to learn task-adaptive features. CTM (Li et al. 2019a) uses concentrator and projector to get a task-level attention map for all samples. However, the differences between samples are ignored. Feat (Ye et al. 2020) follows the similar idea and use a transformer to get the adaptive embedding of the support features. However, such a processing does not take into account the query samples. In this work, we focus on how to produce better feature embedding of images. To be specific, the feature extractor is expected to possess good transferability in instance matching among different tasks rather than only pursing the discrimination of features.

In addition, several works target on graph models. Garcia *et al* (Garcia and Bruna 2018) exploit graph neural networks (GNN) to connect support data and query data and use the features processed by GNN to classify the query. EGNN (Garcia and Bruna 2018) learns to predict the edge-labels that enables the evolution of a clustering by exploiting intra-cluster similarity and inter-cluster dissimilarity.

### Image Inpainting

Image inpainting is to recover the missing regions of an image with plausible synthesized content. Currently, many methods employs the encoder-decoder architecture to map the corrupted image to the complete image. For example, GMCNN (Wang et al. 2018) uses a Generative Multi-column Convolutional Neural Network to generate plausi-

ble synthesized content. CA (Yu et al. 2018) takes a coarse-to-fine structure with contextual attention module. More recently, EC (Nazeri et al. 2019) further improves the quality of generated images by two-stage model with an edge generator followed by an image completion network. Compared with the corrupted images with holes, the inpainted images usually looks natural a lot. In this work, we employ image inpainting to process the erased images, and particularly we use EC due to its superior performance.

# Our Approach

## Problem Formulation

Few-shot classification has three important concepts: large labeled data set $\mathcal{D}_{base}$, support set $\mathcal{S}^{test}$, and query set $\mathcal{Q}^{test}$. The support set contains $N$ categories which do not belong to $\mathcal{D}_{base}$ , and each category contains $M$ samples, namely, $N$-way $M$-shot. The query set $\mathcal{Q}^{test}$ is formed by sampling from the $N$ categories and the samples are not same as that in the support set. Our goal is to use $\mathcal{D}_{base}$ and support set to correctly classify the samples in the query set into the $N$ categories.

For such a learning problem, the episode training mechanism (Vinyals et al. 2016) is proposed and shows excellent performance (Vinyals et al. 2016; Hou et al. 2019; Santoro et al. 2016). The idea of the episode training is to imitate the test process in the training stage, which can maintain the consistency of the training and test process. Specifically, in each episode, we construct a support set $\mathcal{S}^{train}$ and query set $\mathcal{Q}^{train}$ from $\mathcal{D}_{base}$, which are defined as $\mathcal{S}^{train} = \{S^c\}_{c=1}^{N}$ ($|S^c| = M$) and $\mathcal{Q}^{train} = \{Q^c\}_{c=1}^{N}$ ($|Q^c| = M^{'}$), where $c$ is the class index. Let $s_j$ be a sample in $\mathcal{S}^{train}$, where $j$ is the index among all samples. Similarly, $q_i$ represents a sample in $\mathcal{Q}^{train}$. We use an embedding function $g$ to extract features for $s_j$ and $q_i$, resulting in $g_{s_j}$ and $g_{q_i}$. For the support features, we compute a prototype vector for each class using the mean vector as

$$m_c = \frac{1}{|S^c|} \sum_{s_j \in S^c} g_{s_j}, c = 1, 2...N. \tag{1}$$

Given a distance function $d$, then we can calculate the distribution of a query sample over class as

$$p_{i,c} = \frac{\exp(d(g_{q_i}, m_c))}{\sum_c \exp(d(g_{q_i}, m_c))} \tag{2}$$

For convenience, we define $I(*)$ as the label function which gets the ground truth of a sample:

$$I(q_i) = c, \forall q_i \in Q^c (c = 1, 2...N). \tag{3}$$

Then the loss $L$ for the training of this episode is defined by a cross-entropy loss averaged across all query-support pairs, *i.e.*,

$$L = -\frac{1}{NM'} \sum_i \sum_c \mathbf{1}[I(q_i) = c] \log p_{i,c}. \tag{4}$$

The training proceeds by iteratively sampling episodes and minimizing the $L$ to find the suitable feature extractor $g$. After the training is finished, we use $g$ to extract the feature embedding $g_{s_i^t}$ and $g_{q_i^t}$ corresponding to the sample in the support set $\mathcal{S}^{test}$ and query set $\mathcal{Q}^{test}$. Similarly, we calculate the prototype $m_c^t$ for each category in the support set $\mathcal{S}^{test}$ using Equation (1). Finally, we use a nearest neighbors classifier to determine the category of $q_i^t$ as

$$\hat{y}_{q_i^t} = \arg\min_c d\left(g_{q_i^t}, m_c^t\right). \tag{5}$$

It can be seen that the above training procedure is different from the standard classification. For the standard classification, the weight vectors of different classes are directly learned from larger labeled data, and then are used to determine the category of a test sample. In the episode training, however, the mean vector used to determine the category of a sample is calculated by the embedding function trained on other data. That is, the embedding knowledge learned on the base set is required to transfer well to the support and query set for instance matching. In this paper, therefore, we argue that the embedding function need possess good transferability other than extracting discriminative features, *i.e.*, it needs to generate robust feature embedding for unseen classes.

## Approach Overview

In this work, we aim to get better image features for metric calculation in few-shot classification. To this end, we propose to enhance the transferability of embedding function and improve the adaptation of features to the tasks. Figure 2 illustrates the overall architecture of our proposed method. Specifically, a shared embedding function is constructed for all sets that translates the input images into the feature embedding. Then the feature embedding is modulated to adapt the current task so that the final features for distance metric are more specific.

In particular, we propose a novel cross-set erasing-inpainting (CSEI) module to enforce the embedding function to extract intact features representing more complete information of target objects. In principle, CSEI first erases the most discriminative area of an image and then enforces the embedding network to respond over other areas. Consequently, more information of target objects can be extracted. In addition, we propose a task-specific feature modulation (TSFM) module to improve the adaptability of features to the current task.

## Cross-Set Erasing-Inpainting

The cross-set erasing-inpainting module mainly consists of two parts, *i.e.*, erasing-inpainting and cross-set data augmentation. We first use erasing-inpainting to process the support images to produce new images, and then we use cross-set data augmentation to construct a new task. Finally, the model is trained using the new task. Note that CSEI is only used in the training stage and thus does not cause the increase of inference time.

**Erasing-Inpainting** We first need to train a traditional classification network containing a global pooling layer on the base set $\mathcal{D}_{base}$. For each input image $I$, let $f_i(x, y)$ represent the activation of $i$-th feature map before the global

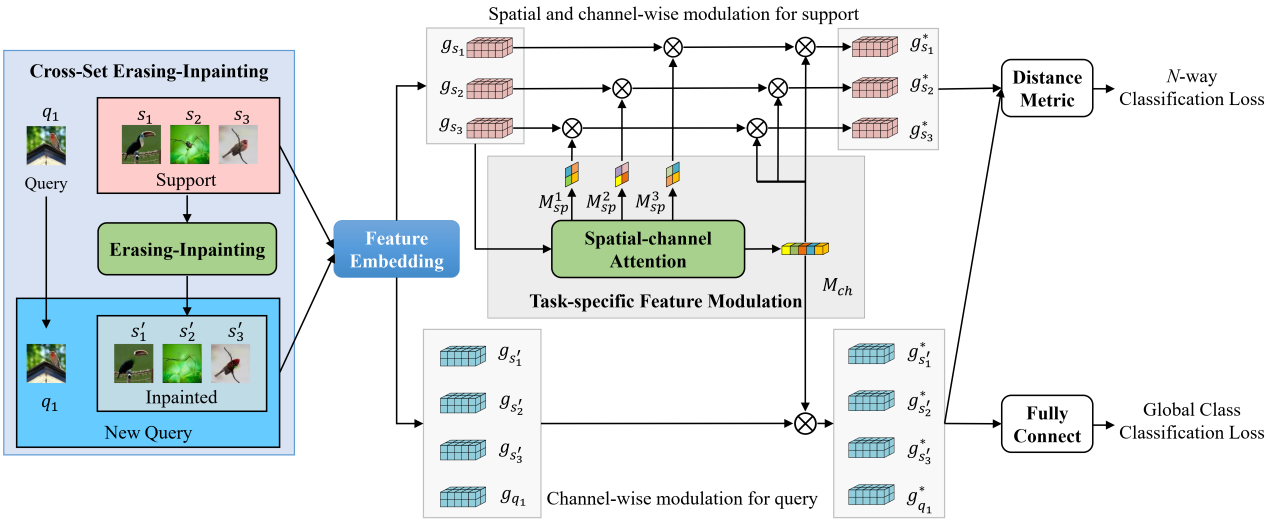



Figure 2: Illustration of our proposed method. We perform cross-set erasing-inpainting (CSEI) on the input images to build a new training task and use task-specific feature modulation (TSFM) to enforce the features for metric calculation adaptive to the current task. Best viewed in color.

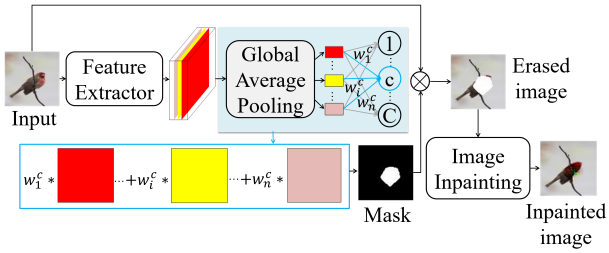



Figure 3: Illustration of our proposed erasing-inpainting module. Best viewed in color.

pooling layer at the location $(x, y)$. Then the score of a category $c$ is $O_c = \frac{1}{HW}\sum_i w_i^c \sum_{x,y} f_i(x, y)$, where $H$ and $W$ denote the spatial resolution and $w_i^c$ is the weight corresponding to the class $c$ for the unit $i$. We can get the class activation map $M_c$ for the class $c$ with the elements

$$M_c(x, y) = \sum_i w_i^c f_i(x, y). \tag{6}$$

As stated in (Zhou et al. 2016), $M_c(x, y)$ indicates the importance of the activation at the spatial grid $(x, y)$ to classify the image $I$ into the class $c$. Therefore, $M_c(x, y)$ can serve as the indicator to the most discriminative regions for the class $c$. More specifically, we first sort the scores for each image and get the index of the top $K$ categories with the highest scores $(O_{c_1}, .., O_{c_k}, ..., O_{c_K})$. Then we can get $K$ class activation maps $(M_{c_1}, ..., M_{c_k}, ..., M_{c_K})$.

We normalize $M_{c_k}$ to $(0, 1)$ and then use a threshold strategy to convert it into a binary mask $\bar{M}_{c_k}$. Then we get the masked images $\bar{I}_k = I \odot (1 - \bar{M}_{c_k})$, where $\odot$ denotes the spatially element-wise multiplication, which erases the discriminative regions of the input image. Here the erased regions are assigned the value 0. Evidently, the erased region is very inconsistent with the surrounding region and it will bring the interference. In this work, we use the image inpainting technique to complete the erased images such that the erased area is consistent with the surrounding area. In the experiments, we directly use EC (Nazeri et al. 2019) model provided by the authors to complete $\bar{I}_k$ and get an inpainted image $\bar{I}_k^{'}$. Consequently, for each image $I$ sampled from $\mathcal{D}_{base}$, we can get $K$ inpainted images $\bar{I}_1^{'}, ...\bar{I}_k^{'}, ..., \bar{I}_K^{'}$. In practice, we process all images in $D_{base}$ in advance and store the results on the disk for reducing the training time.

**Cross-Set Data Augmentation** In this work, we use the images produced by erasing-inpainting to enhance the feature extractor via data augmentation. Specifically, there are a support set $\mathcal{S}^{train}$ and a query set $\mathcal{Q}^{train}$ for each task. We use the erasing-inpainting module to process the images in the support set $\mathcal{S}^{train}$ and get a set of inpainted images $\mathcal{S}_{EI}^{train} = \{S_{EI}^c\}_{c=1}^N$, where $|S_{EI}^c| = MK$. Then we can build a new query set by uniting the original query set and inpainted support set, *i.e.*,

$$\mathcal{Q}_{new}^{train} = \mathcal{Q}^{train} \cup \mathcal{S}_{EI}^{train}. \tag{7}$$

Finally, we use the task $(\mathcal{S}^{train}, \mathcal{Q}_{new}^{train})$ to train our model with the episode training strategy.

It can be seen that we use the samples in the support set to augment the query set, which thus is called *cross-set data augmentation*. Through such data processing, the feature extractor is enforced to respond more regions of target objects other than the discriminative part. To be specific, the inpainted sample in the query set has been erased the discriminative part (*e.g.*, "head"), and the corresponding feature would not contain the information of that part. During the training, in order to match the query sample to the support samples correctly, the feature extractor would have to capture the information outside the discriminative part.

Actually, we have another choice to augment the query set, named *same-set data argumentation*. That is, we use the erasing-inpainting module to process the images in the query set $\mathcal{Q}^{train}$, and then build a new query set by uniting the original and inpainted query sets. For the same-set data argument, the support and query images are still different but increasing the size of query set. Obviously, this cannot enforce the feature extractor to capture more object information.

### Task-Specific Feature Modulation

For different tasks, the features to distinguish the involved classes may be vastly different even for the same image. That is, the features for metric calculation need adapt to the current task. To this end, we propose to modulate the feature embedding from two aspects, as shown in Figure 2. First, we learn a channel-wise attention $M_{ch}$ for both the support and query sets to highlight the important patterns since each channel of the feature maps represent a certain pattern. Second, we learn a spatial attention for only the support set to highlight the target objects, where different attention maps are produced for samples in the support set.

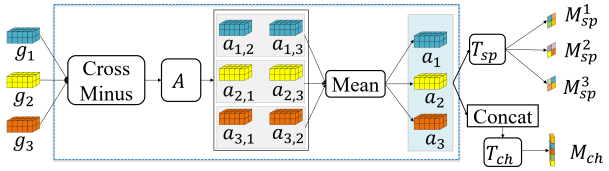



Figure 4: Illustration of our proposed task-specific feature modulation. Here $A$, $T_{sp}$, and $T_{ch}$ represent three nonlinear transformation.

Formally, assume that we have gotten the prototypes $M_s = (m_1, \cdots, m_c, \cdots, m_N)$ for each category in the support set and the query features $\mathcal{G}_q = (g_{q_1}, \cdots, g_{q_{NM'}})$. Then we fuse the information of all the samples in the support set to learn the attention maps, as shown in Figure 4. Here the support samples actually represent the current task. To be specific, we first use the *cross minus* to calculate the difference between prototypes and then apply a nonlinear transformation $A$ to get $a_{i,j} = A(m_i - m_j)$. After that, we get the difference information between the features of a certain category $c$ and that of other categories, *i.e.*,

$$a_c = -\frac{1}{N-1}\sum_{j \neq c} a_{c,j}. \tag{8}$$

One hand, we concatenate the difference information $\{a_c\}_{c=1}^{N}$ along the channel, and then use a nonlinear transformation $T_{ch}$ to get the channel-wise attention vector:

$$M_{ch} = T_{ch}(([a_1, a_2, ..., a_N])). \tag{9}$$

On the other hand, we use a nonlinear transformation $T_{sp}$ to compress the channels of $a_c$ into 1-dimension to get a spatial attention for each prototype $c$, *i.e.*,

$$M^c_{sp} = T_{sp}(a_c). \tag{10}$$

Finally, we use $M^c_{sp}$ and $M_{ch}$ to modulate the features for the support and query sets as

$$m^*_c = m_c \odot (1 + M^c_{sp}) \odot (1 + M_{ch}), \tag{11}$$

$$g^*_{q_i} = g_{q_i} \odot (1 + M_{ch}), \tag{12}$$

where $\odot$ denotes the element-wise multiplication along the spatial position or channels. Note that we only apply the task-level channel-wise attention to the query samples, which indeed represents the discriminative patterns of the current task.

### Loss Function

We build a $N$-way classification loss $L_{N-way}$ as in Equation (4), and meanwhile build a global classification loss $L_{global}$ that is appended to the fully connected layer to classify the features $g^*_{q_i}$ into all categories in $\mathcal{D}_{base}$. Then the overall loss function is defined as

$$L_{total} = L_{global} + \lambda * L_{N-way}. \tag{13}$$

Here, we set $\lambda$=0.5 as same as CAN (Hou et al. 2019). In addition, both $L_{global}$ and $L_{N-way}$ adopt the form of dense classification like CAN.

## Experiments

### Experiment Setup

**Dataset description.** Following the previous works (Hou et al. 2019; Li et al. 2019a; Ye et al. 2020), we use *miniImageNet* (Vinyals et al. 2016) and *tiredImageNet* (Ren et al. 2018) to evaluate the performance of different methods. The miniImageNet consists of 100 classes with 600 images per class. According to the standard split (Vinyals et al. 2016), the 100 classes are divided into 64 training classes, 16 validation classes, and 20 test classes, respectively. The tiredImageNet is a larger dataset, which totally consists of 34 categories (608 classes), and have 779, 165 images for training, validation, and test. The 34 categories (608 classes) are divided into 20 training categories (351 classes), 6 validation categories (97 classes), and 8 test categories (160 classes).

**Implementation details.** We use ResNet-12 as the feature extractor, and stack three convolutional layers to construct the nonlinear transformations $A$ and $T_{sp}$ in TSFM. As for $T_{ch}$, we use a convolutional layer and a SElayer (Hu, Shen, and Sun 2018) to generate the channel-wise attention. Following the previous works, we resize the input images into $84 \times 84$, and some typical data augmentation strategies (*e.g.*, random crop and erase) are employed. In the training stage, we first train the feature extractor with CSEI on $\mathcal{D}_{base}$. Then we retrain the model equipped with both CSEI and TSFM. For *miniImageNet*, the model is trained for 110 epochs, each of which contains 1, 200 episodes. For *tiredImageNet*, the model is trained for 100 epochs, each of which contains 13, 980 episodes. Particularly, for each training episode, we randomly sample 30 query samples. In the test stage, each episode consists of 75 query samples. We report the performance of different methods using the mean accuracy as well as the 95% confidence interval on 600 randomly generated episodes. PyTorch (Paszke et al. 2017) and NVIDIA 1080Ti GPUs are used throughout our experiments.

| Method | Type | Backbone | miniImageNet | | tieredImageNet | |
|---|---|---|---|---|---|---|
| | | | 5-way 1-shot | 5-way 5-shot | 5-way 1-shot | 5-way 5-shot |
| LEO (Rusu et al. 2019) | **O** | WRN-28 | $61.76 \pm 0.08$ | $77.59 \pm 0.12$ | $66.33 \pm 0.05$ | $81.44 \pm 0.09$ |
| MetaOpt (Lee et al. 2019) | **O** | ResNet-12 | $62.64 \pm 0.62$ | $78.63 \pm 0.46$ | $65.99 \pm 0.72$ | $81.56 \pm 0.53$ |
| RestoreNet (Xue and Wang 2020) | **O** | ResNet-18 | $59.28 \pm 0.20$ | - | - | - |
| IFSL (Yue et al. 2020) | **O** | WRN-28 | 64.40 | 80.20 | 71.45 | 86.02 |
| MatchingNet (Vinyals et al. 2016) | **M** | ConvNet | $43.44 \pm 0.77$ | $60.60 \pm 0.71$ | - | - |
| CTM (Li et al. 2019a) | **M** | ResNet-18 | $64.12 \pm 0.82$ | $80.51 \pm 0.13$ | $68.41 \pm 0.39$ | $84.28 \pm 1.73$ |
| TapNet (Yoon, Seo, and Moon 2019) | **M** | ResNet-12 | $61.65 \pm 0.15$ | $76.36 \pm 0.10$ | $63.08 \pm 0.15$ | $80.26 \pm 0.12$ |
| DSN (Simon et al. 2020) | **M** | ResNet-12 | $64.60 \pm 0.72$ | $79.51 \pm 0.50$ | $67.39 \pm 0.82$ | $82.85 \pm 0.56$ |
| FEAT (Ye et al. 2020) | **M** | ResNet-12 | 66.78 | 82.05 | $70.80 \pm 0.23$ | $84.79 \pm 0.16$ |
| $E^3$BM (Liu, Schiele, and Sun 2020) | **M** | ResNet-25 | 64.3 | 81.0 | 70.0 | 85.0 |
| EMD-FCN (Zhang et al. 2020b)$^+$ | **M** | ResNet-12 | $65.64 \pm 0.28$ | $81.64 \pm 0.57$ | $71.00 \pm 0.32$ | $85.01 \pm 0.67$ |
| CAN (Hou et al. 2019)$^+$ | **M** | ResNet-12 | $64.42 \pm 0.84$ | $79.03 \pm 0.58$ | $70.65 \pm 0.99$ | $84.08 \pm 0.68$ |
| **Ours+CAN** | **M** | ResNet-12 | $\mathbf{67.59 \pm 0.83}$ | $\mathbf{81.93 \pm 0.36}$ | $\mathbf{72.57 \pm 0.95}$ | $\mathbf{85.72 \pm 0.63}$ |
| EMD-9s (Zhang et al. 2020a)$^+$ | **M** | ResNet-12 | $67.82 \pm 0.27$ | $84.12 \pm 0.52$ | $73.08 \pm 0.33$ | $86.93 \pm 0.62$ |
| **Ours+EMD-9s** | **M** | ResNet-12 | $\mathbf{68.94 \pm 0.28}$ | $\mathbf{85.07 \pm 0.50}$ | $\mathbf{73.76 \pm 0.32}$ | $\mathbf{87.83 \pm 0.59}$ |

Table 1: Performance comparison with the state-of-the-arts. Here we retrain CAN and EMD-* using the codes provided by authors for fair comparison, which is denoted by $+$. For IFSL, we report the highest result under the inductive setting. The methods are divided into two groups, *i.e.*, optimization-based methods (**O**) and metric-based methods (**M**).

## Comparison with the State-of-The-Art

In this section, we report the performance of different models in Table 1, which contains the optimization-based methods and metric-based methods under inductive setting. Here we particularly use the CAN as the main baseline, and meanwhile evaluate our proposed modules on the recent EMD-9s. Particularly, for EMD-9s, it randomly crops 9 patches with different sizes and shapes from the original images and then resizes these patches to $84 \times 84$ for test. For other experiments, we directly resize the original image to $84 \times 84$ for test. For these baseline methods, we need to retrain the models using the codes provided by the authors for fair comparison.

From the results in Table 1, we have the following observations. First, our method always can bring the accuracy increase compared with the corresponding baseline, which demonstrates the effectiveness of our proposed techniques. Second, when combined with EMD-9s, our method achieves the state-of-the-art performance on both datasets. However, the improvement of our method on EMD-9s is lower than that on CAN. The reason is that for EMD-9s, the spatial information of the support and query samples are reserved by local features. This would alleviate the problem that the model responds to local areas. Third, our method usually can make larger gains for 1-shot than 5-shot. For example, more $3\%$ increase of average accuracy is achieved for the 5-way 1-shot of CAN on miniImageNet. It also implies that extracting the intact features are important for few-shot learning.

In addition, we particularly calculate the intra-class variances and inter-class variances on the test set of miniImageNet for CAN and our method , i.e., 1.234 vs. 0.864 for the intra-class variances, and 0.958 vs. 0.959 for the inter-class variances. It can be seen that our method hardly changes the inter-class variance while reducing the intra-class variance. This makes the features more distinguishable.

| CSEI | | TSFM | | miniImageNet | |
|---|---|---|---|---|---|
| CSE | inpaint | spatial | channel | 1-shot | 5-shot |
| | | | | 64.42 | 79.03 |
| ✓ | | | | 66.18 | 80.88 |
| ✓ | ✓ | | | 66.63 | 81.32 |
| ✓ | ✓ | ✓ | | 66.91 | 81.46 |
| ✓ | ✓ | | ✓ | 66.89 | 81.70 |
| ✓ | ✓ | ✓ | ✓ | **67.59** | **81.93** |

Table 2: Effect of different modules.

## Ablation Study

In this section, we deeply investigate the proposed method. Particularly, we conduct experiments on the miniImageNet dataset and use CAN as the baseline.

**Effect of different components.** Here we investigate the effect of our proposed components on the performance by evaluating their combinations. Particularly, we consider the CSEI and its no-inpainting version (CSE) that the erased images are directly used. As for TSFM, spatial and channel-wise attentions are separately explored. Table 2 gives the results, where the first row denotes the baseline. We have the following observations. First, the accuracy gain mainly comes from CSEI and TSFM can further bring an increase. Second, image inpainting can improve the performance compared with directly using erased images. Third, both spatial attention and channel-wise attention can bring performance gains, and their combination performs best.

**CSEI analysis.** We conduct more experiments to deeply understand the CSEI module, where $K = 4$ without TSFM is particularly used for efficiency. In form, CSEI processes the data in the support set to augment the query set. For contrast, we consider another two ways to augment the query set: a) directly sampling a larger query set to keep the same

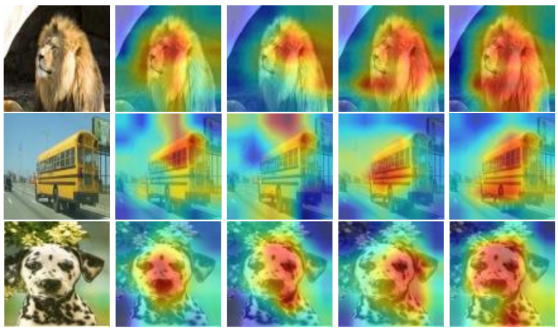



Figure 5: Activation maps for different methods: (a) input, (b) baseline, (c) Query50, (d) SSEI, and (e) CSEI. Here warmer color indicates higher values. Best viewed in color.

| Model | miniImageNet | |
|---|---|---|
| | 1-shot | 5-shot |
| Baseline | 64.42 | 79.03 |
| Query50 | 64.46 (+0.04) | 78.88 (-0.15) |
| SSEI (same-set) | 64.88 (+0.46) | 79.13 (+0.10) |
| CSEI (cross-set) | 66.63 (+2.21) | 81.32 (+2.29) |

Table 3: Comparison of different augmentation methods.

size as ours for each training episode (50 query samples for 5-way 1-shot), and b) conducting the erasing-inpainting randomly over the query samples, *i.e.*, same-set erasing-inpaiting (SSEI), to keep the same size. Table 3 gives the results. It can be seen that simply enlarging the query set cannot bring performance gain while erasing-inpainting can increase the accuracy. Furthermore, CSEI significantly outperforms SSEI. To more intuitively understand such a result, Figure 5 gives the activation maps of some samples to compare different augmentation methods. It can be seen that CSEI can better highlight the whole area of target objects.

**Hyper-parameter $K$.** In CSEI, the hyper-parameter $K$ is used to control the number of erasing-inpainting images for an image in the support set. Here we explore its effect on classification performance and particularly evaluate the

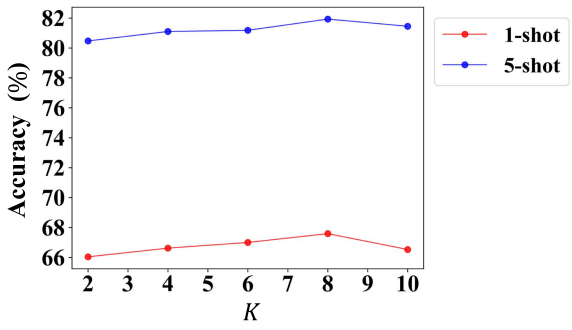



Figure 6: Results of different hyper-parameter $K$ on miniImageNet. In our experiment, $K = 8$ is adopted.


Figure 7: Erased images of a sample. Here the class with the $k$-th highest score is used to produce the erased image.

| Method | 1-shot | | 5-shot | |
|---|---|---|---|---|
| | baseline | ours | baseline | ours |
| Proto-net | 61.44 | 63.24 | 76.42 | 78.66 |
| DC | 63.69 | 65.97 | 78.14 | 80.63 |
| EMD-FCN | 65.64 | 66.60 | 81.64 | 82.50 |

Table 4: Effect of our proposed method for different baseline methods on miniImageNet.

value from $\{2, 4, 6, 8, 10\}$. Figure 6 shows the results on miniImageNet. It can be seen that $K = 8$ gets the best performance for both *5-way 1-shot* and *5-way 5-shot*, and a larger $K$ would slightly decrease the accuracy. To intuitively understand the results, Figure 7 shows the erased images of a sample, where the classes with the $k$-th highest score are used separately. It can be seen that different regions are located by different classes. In particular, a small value of $k$ usually makes the erased region focus on the target objects. But a larger value may locate the background regions, which would hurt the performance.

**Generalization of our method.** Our proposed CSEI and TSFM can be applied to the off-the-shelf methods of few-shot learning. Here we integrate them into more methods to verify the generalization of our method. In particular, the Proto-net (Snell, Swersky, and Zemel 2017), DC (Lifchitz et al. 2019), and EMD-FCN (Zhang et al. 2020b) are particularly employed. For Proto-net and DC, we use ResNet-12 as the feature extractor to re-implement these methods. For EMD-FCN, we retrain it using the codes provided by the authors. Table 4 gives the results, and it is demonstrated that our method can consistently bring the performance gain for different baseline methods.

## Conclusion

In this paper, we argue that the transferability of the embedding function in instance matching is very important for few-shot learning, and then propose a novel cross-set erasing-inpainting (CSEI) method that achieves such a goal by pursuing to represent whole target objects. To be specific, CSEI first erases the discriminative regions of an image in the support set, and then uses the inpainted images to augment the query set. Consequently, CSEI would enforce the feature embedding to contain more information of target objects rather than only the most discriminative part. In addition, we propose task-specific feature modulation (TSFM) to adapt image features to the current task. We experimentally verified the effectiveness of our proposed methods, which can consistently improve the classification performance for different baseline methods.

## Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant 61673362 and 61836008, Youth Innovation Promotion Association CAS (2017496). In addition, we acknowledge the support of GPU cluster built by MCC Lab of Information Science and Technology Institution, USTC.

## References

Antoniou, A.; Edwards, H.; and Storkey, A. 2018. How to train your MAML. *ICLR* .

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.

Garcia, V.; and Bruna, J. 2018. Few-shot learning with graph neural networks. In *ICLR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hou, R.; Chang, H.; Bingpeng, M.; Shan, S.; and Chen, X. 2019. Cross attention network for few-shot classification. In *NeurIPS*.

Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitation networks. In *CVPR*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NeurIPS*.

Lee, K.; Maji, S.; Ravichandran, A.; and Soatto, S. 2019. Meta-learning with differentiable convex optimization. In *CVPR*.

Li, H.; Eigen, D.; Dodge, S.; Zeiler, M.; and Wang, X. 2019a. Finding task-relevant features for few-shot learning by category traversal. In *CVPR*.

Li, W.; Wang, L.; Xu, J.; Huo, J.; Gao, Y.; and Luo, J. 2019b. Revisiting local descriptor based image-to-class measure for few-shot learning. In *CVPR*.

Li, X.; Sun, Q.; Liu, Y.; Zhou, Q.; Zheng, S.; Chua, T.-S.; and Schiele, B. 2019c. Learning to self-train for semi-supervised few-shot classification. In *NeurIPS*.

Lifchitz, Y.; Avrithis, Y.; Picard, S.; and Bursuc, A. 2019. Dense classification and implanting for few-shot learning. In *CVPR*.

Liu, Y.; Schiele, B.; and Sun, Q. 2020. An Ensemble of Epoch-wise Empirical Bayes for Few-shot Learning. *ECCV* .

Munkhdalai, T.; Yuan, X.; Mehri, S.; and Trischler, A. 2018. Rapid adaptation with conditionally shifted neurons. In *ICML*.

Nazeri, K.; Ng, E.; Joseph, T.; Qureshi, F.; and Ebrahimi, M. 2019. EdgeConnect: Structure Guided Image Inpainting using Edge Prediction. In *ICCV workshop*.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch. *NeurIPS workshop* .

Ren, M.; Triantafillou, E.; Ravi, S.; Snell, J.; Swersky, K.; Tenenbaum, J. B.; Larochelle, H.; and Zemel, R. S. 2018. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676* .

Rusu, A. A.; Rao, D.; Sygnowski, J.; Vinyals, O.; Pascanu, R.; Osindero, S.; and Hadsell, R. 2019. Meta-learning with latent embedding optimization. *ICLR* .

Santoro, A.; Bartunov, S.; Botvinick, M.; Wierstra, D.; and Lillicrap, T. 2016. Meta-learning with memory-augmented neural networks. In *ICML*.

Simon, C.; Koniusz, P.; Nock, R.; and Harandi, M. 2020. Adaptive Subspaces for Few-Shot Learning. In *CVPR*.

Snell, J.; Swersky, K.; and Zemel, R. 2017. Prototypical networks for few-shot learning. In *NeurIPS*.

Sun, Q.; Liu, Y.; Chua, T.-S.; and Schiele, B. 2019. Meta-transfer learning for few-shot learning. In *CVPR*.

Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P. H.; and Hospedales, T. M. 2018. Learning to compare: Relation network for few-shot learning. In *CVPR*.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *CVPR*.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *CVPR*.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. 2016. Matching networks for one shot learning. In *NeurIPS*.

Wang, Y.; Tao, X.; Qi, X.; Shen, X.; and Jia, J. 2018. Image Inpainting via Generative Multi-column Convolutional Neural Networks. In *NeurIPS*.

Wei, Y.; Feng, J.; Liang, X.; Cheng, M.-M.; Zhao, Y.; and Yan, S. 2017. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *CVPR*.

Xue, W.; and Wang, W. 2020. One-Shot Image Classification by Learning to Restore Prototypes. *AAAI* .

Ye, H.-J.; Hu, H.; Zhan, D.-C.; and Sha, F. 2020. Few-shot learning via embedding adaptation with set-to-set functions. In *CVPR*.

Yoon, S. W.; Seo, J.; and Moon, J. 2019. Tapnet: Neural network augmented with task-adaptive projection for few-shot learning. *ICML* .

Yu, J.; Lin, Z.; Yang, J.; Shen, X.; Lu, X.; and Huang, T. S. 2018. Generative Image Inpainting with Contextual Attention. *CVPR* .

Yue, Z.; Zhang, H.; Sun, Q.; and Hua, X.-S. 2020. Interventional few-shot learning. *NeurIPS* .

Zhang, C.; Cai, Y.; Lin, G.; and Shen, C. 2020a. DeepEMD: Differentiable Earth Mover's Distance for Few-Shot Learning. *arXiv preprint arXiv:2003.06777v3* .

Zhang, C.; Cai, Y.; Lin, G.; and Shen, C. 2020b. DeepEMD: Few-Shot Image Classification With Differentiable Earth Mover's Distance and Structured Classifiers. In *CVPR*.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. In *CVPR*.