Letting 3D Guide the Way: 3D Guided 2D Few-Shot Image Classification

Jiajing Chen, Minmin Yang, Senem Velipasalar
Syracuse University, Electrical Engineering and Computer Science Dept., Syracuse, NY, USA
{jchen152, myang47, svelipas}@syr.edu [*]

Abstract

Existing few-shot image classification networks aim to perform prediction on images belonging to classes that were not seen during training, with only a few labeled images, which are randomly picked from the same image pool as the support set. However, this traditional approach has two main issues: (i) in real-world applications, since support images are randomly picked, the angle they were captured from can be very different from that of the query image, causing the images to look very different and making it hard to match them; (ii) since support and query images, for both training and testing, are sampled from the same image pool, models can overfit the dataset, especially if the image pool contains images with similar color, texture or view angle. Thus, good performance on a dataset does not reflect a model's real ability. To address these issues, we propose a novel few-shot learning approach referred to as the 3D guided 2D (3DG2D) few-shot image classification. In our proposed approach, the queries are 2D images, and the support set is composed of 3D mesh data, providing different views of an object, in contrast to randomly picked images providing a single view. From each 3D mesh, 14 projection images are generated from different angles. Thus, these projections have significant variance among themselves. To address this challenge, we also propose the Angle Inference Module (AIM), which is used to infer the view angle of a query image so that more attention is given to projection images corresponding to the same view angle as the query image to achieve better prediction performance. We perform experiments on ModelNet40, Toys4K and ShapeNet datasets with 4-fold cross validation, and show that our 3DG2D few-shot classification approach consistently outperforms the state-of-the-art baselines.

1. Introduction

Recent years have witnessed significant developments in supervised learning tasks, such as 2D image classification, object detection and segmentation, thanks to Convolutional Neural Networks (CNNs). However, to perform well, these supervised-learning networks need to be trained with large amounts of labeled data. Annotation of large datasets requires significant amount of human time and labor, and depending on the application, very large amounts of data may not be available or may be hard to collect.

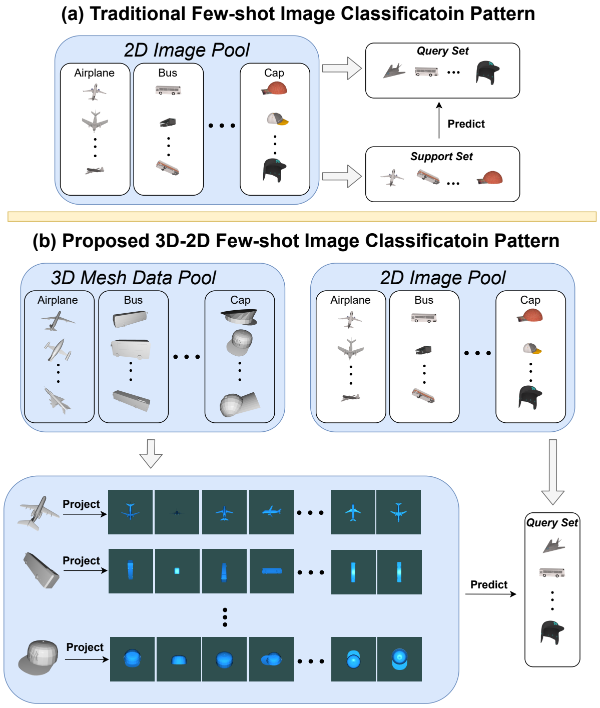

Figure 1. (a) Traditional few-shot image classification approach, wherein support and query images are randomly sampled from the same image pool; (b) our proposed 3DG2D few-shot image classification approach. Instead of randomly picking 2D images as support data, for each 3D mesh sample, we generate 14 projection images from different angles.

On the other hand, humans have more advanced learning abilities, considering that they can learn to recognize never-before-seen objects from only a few examples. Humans perceive the world and interact with objects in 3D, and this plays an important role in building 'prior' knowl-

[*] This work was supported in part by the National Science Foundation (NSF) under Grant 2221875 and New York State Department of Economic Development under Contract Number C080116.

edge [5,7]. Seeing an object in 3D allows observing it from different views, and eliminates the need for seeing a large number of 2D images of this object taken from different angles. Moreover, humans are very good at learning and identifying similarities and differences between objects, which is one of the main ideas behind few-shot learning.

In N-way K-shot Q-query image classification task, all classes in the dataset are split into two sets of classes: C_{train} and C_{test}, for training and testing, respectively. Since few-shot learning (FSL) aims to perform prediction on objects belonging to never-before-seen classes, $C_{train} \cap C_{test} = \varnothing$. For both the training and testing stages, N classes are randomly selected from C_{train} and C_{test}, respectively. Then, for each class, K samples are picked and deposited into the support set, and Q samples are picked and placed into the query set. The labels of query samples are then predicted by matching the samples in the query set with the labeled samples in the support set.

Existing few-shot image classification approaches [10, 11, 13, 15, 20, 23] only use 2D images as support and query images, which are chosen from the same image pool. This traditional approach, which is illustrated in Fig. 1 (a), has **two main issues**: (i) since support images are randomly picked, in a real-world application, the angle they were captured from can be very different from that of the query image, causing the images to look very different and making it hard to match them; (ii) since support and query images, for both training and testing, are sampled from the same image pool, models can overfit the dataset, especially if the image pool contains images with similar color, texture or view angle. Thus, good performance on a dataset does not reflect a model's real ability.

The aforementioned issues with the existing few-shot image classification have not been sufficiently explored by incorporating 3D knowledge. Thus, to address these issues and motivated by the strengths of humans' learning abilities, which are mentioned above, we propose a novel few-shot learning approach referred to as the 3D guided 2D (3DG2D) few-shot image classification. As shown in Fig. 1(b), in our proposed approach, the queries are 2D images, and the support set is composed of 3D mesh data. From each 3D mesh sample, we generate 14 projection images from different view angles. To make things more realistic and challenging, we make sure that the query 2D projection image comes from a mesh sample that is different from the mesh samples used for the support set. In addition, projection images in the support set and query images have different foreground and background colors, and texture and are from different view angles. Since the 14 projections in the support set provide a more complete information about an object, if the query and support images belong to objects from the same class, a match will be much more likely to one of these 14 projections. Initially, without loss of generality,we use the same 2D query images as in [14], since data is already available. As shown in our supplementary material, we also performed experiments, wherein the user only provides 2D RGB images as queries, and we use an existing set of 3D meshes to form a support library of projection images from different angles.

Our proposed approach of using 3D meshes and obtaining 14 projection images for guidance brings up new challenges of its own. Since these 14 projections are generated from different angles, they have significant variance among themselves, and many of these projections will look very different from the query image despite belonging to the same object class. Thus, treating all the shots of a class equally will introduce noise and corrupt the representation. To address this challenge, we also propose the Angle Inference Network (AINet), which is used to infer the view angle of a query image so that more attention is given to projection images corresponding to the same view angle, as the query image, to achieve better prediction performance. The main contributions of this work include the following:

- To address the issues with the traditional few-shot learning approaches, we propose a 3D Guided 2D (3DG2D) few-shot image classification method, which, to the best of our knowledge, is the first work that uses 3D data as the support set to perform 2D image classification.
- We perform experiments to show how the angle variety and shape variety affect the few-shot image classification performance. Angle variety refers to having images of the same object from different viewing angles in the support set, while shape variety refers to objects from the same class with different shapes (e.g. different shaped planes, birds, etc.)
- We perform detailed analysis showing that projections generated from different view angles contribute differently to the classification of query images during testing.
- We propose AINet. Different from most existing methods, which treat all images in the support set equally, AINet first infers the query images' view angle, and then places more attention on support projections obtained from the same angle.
- Experiments performed on ModelNet40, Toys4k and ShapeNet datasets, with 4-fold cross validation, show that our proposed 3DG2D approach using AINet consistently outperforms SOTA methods in terms of average accuracy.

The code is avaliable in `https://github.com/jiajingchen113322/3DG2D.git`.

2. Related Work

Existing few-shot image classification models can be broadly classified into three categories [11]:

i) Learning Feature Embeddings: These methods focus on designing networks that can learn similarities and differentiating features, which can generalize well to unseen classes. Siamese Network [2], which is an earlier work, consists of two sub-networks with identical network structures and shared parameters. Matching Network [18] obtains support and query features by using different networks, and introduces episodic training for few-shot image classification. Stojanov et al. [14] use 3D data to guide a 2D backbone network and learn better support and query embeddings for final prediction. Although this approach makes use of 3D data, it still follows a traditional few-shot image classification flow, and performs query image classification by using 2D images in the support set. Other works [8, 19, 26] adopt a data augmentation strategy to learn better feature embeddings.

ii) Learning Class Representations: Methods in this category propose to use class prototypes, which serve as reference features, for query image classification. ProtoNet [13], a classical few-shot classification method, obtains prototypical features by averaging the support features, which are obtained from the 2D support images belonging to the same class. In other words, ProtoNet treats all the support images belonging to the same class equally. Infinite Mixture Prototypes (IMP) [1], different from ProtoNet, form multiple clusters representing each class. Ravichandran et al. [12] present an approach, wherein a prototype is a learnable, parameterized function of feature embedding.

iii) Learning Distance/Similarity Measures: Models in this category employ different metrics to measure the similarity between support and query features. DeepBDC [24] adopts Brownian Distance Covariance [16, 17] as a similarity measure. FRN [21] reconstructs query features from support features, instead of calculating the distance between support and query features. DeepEMD [25] uses Earth Mover's Distance (EMD) as the distance metric.

3. Motivation

We first study the effect and importance of angle and shape variation for few-shot image classification in Sec. 3.1. Motivated by the observations in Sec. 3.1, we propose our 3DG2D few-shot image classification approach. Then, we show how the support projection images obtained from different view angles contribute differently to classification performance in Sec. 3.2. Based on this observation, we propose the AINet in Sec. 4.2, to give more weight to the projections images in the support set, which were obtained from similar view angles as the query images. All the experiments in this section are performed by using ResNet as the backbone. Similar to ProtoNet, the average of all projection features, in the support set of a class, is computed to obtain the class 'prototype'. The query label is predicted by finding the most similar prototype feature to the query feature.

3.1. Angle and Shape Variation Analysis

We perform a set of experiments on the ShapeNet dataset to study how the angle and shape variety affect the few-shot image classification performance. Angle variety refers to having images of the same object from different viewing angles in the support set, while shape variety refers to having objects from the same class with different shapes. (e.g. different shaped planes, birds, etc.) ShapeNet contains 52K 3D mesh samples from 55 categories. For each category, we pick 50 mesh samples to build a 3D support pool. Fig. 2 (a) shows how projections images are obtained from a 3D mesh. H_1 refers to the projection obtained from a camera placement in front of the object. Each time, the camera is rotated by 90°on the horizontal plane to obtain H_2, H_3 and H_4. T_i and B_i are obtained by moving the camera up and down for 45°, respectively, from the H_i position, where $i \in \{1, ...4\}$. To have more coverage of the 3D object, an additional top view T_5 and bottom view B_5 are obtained, bringing the total number of projections to 14 for each 3D mesh. As for the query image pool, instead of generating the images ourselves, we use the images from ShapeNet-LS [14]. To simulate a more realistic scenario, the support projections and query images are generated from separate 3D object sets with no overlap, and they have different color, texture and view angle. Sample query images are shown in the 2D Image Pool in Fig. 1 (b).

55 classes in ShapeNet are first sorted by their class ID, and then divided into four folds, with 14, 14, 14 and 13

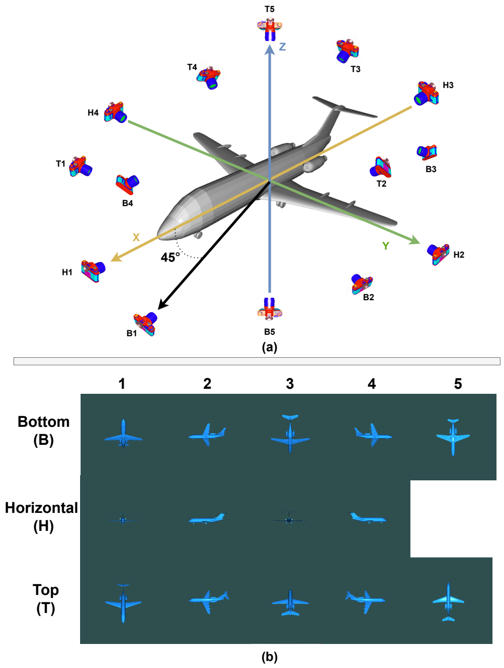

Figure 2. (a) 14 camera positions are used to generate the projection images shown in (b).

classes in each fold. Classes in the first fold are used for testing while the rest is used for training. We perform 5-way K-shot 10-query experiment with projections used as support images. To analyze the effects of angle and shape variety, the following strategies are used to pick the projection images for the support set:

- S1. Angle variety for a fixed shape: A 3D object is picked first. Then, K projections belonging to this object are used as support images. As K is increases, more angle variety is covered in the support set.
- S2. Shape variety for a fixed angle: An angle is picked first. Then, K projections taken from that angle but for different samples from the same class are used as support.
- S3. Mixed variety: K projections are randomly picked from the projection pool resulting in images representing different view angles and as well as different shapes.

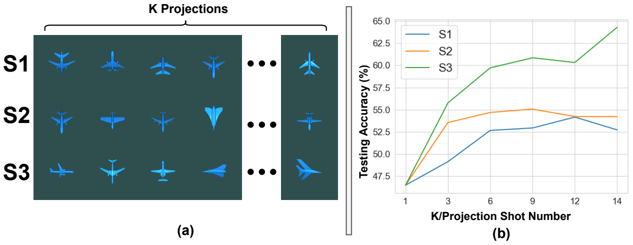

Figure 3. (a) Sample support projections under three different strategies; (b) The plot of accuracy versus number of images in the support set for different strategies of building the support set.

Fig. 3.1(a) shows the sample support images chosen with strategies S1,S2 and S3. We perform the few-shot image classification experiment with different K values. From the accuracy plots, shown in Fig. 3.1(b), following conclusions can be drawn: (i) Increasing the angle variety (blue curve) improves the accuracy in general. Same is true for the shape variety (orange curve); (ii) If only angle variety or shape variety is increased, when K is 9-12, the accuracy converges on a similar peak value (53%-55%); (iii) Having shape variety seems to contribute more to final accuracy than angle variety (orange curve is always above the blue one), and this is expected considering different shapes an object class can take; (iv) When support set has more variety (increasing K) both in terms of angles and shape (green curve), the accuracy is further boosted.

Existing few-shot learning approaches pick up 2D support images randomly, without paying special attention to having angle variety or shape variety in a support set. Shape variety can only be achieved by increasing the shot number. However, angle variety can be obtained from different projections of just one 3D mesh sample.

3.2. Analysis of View Angle Contribution

A 3D object can look different from different view angles. For example, in Fig. 2 (b), projections H_1 and H_3 look very different from other projections. A query image will be a closer match to one of these projections. In other words, contribution of different view angles is different, and dependent on view angle distribution of the query set. To analyze this, we perform a 5-way 1-shot/projection 10 query experiment on ModelNet40 and ShapeNet datasets. ModelNet40 contains 3D mesh data from 40 categories. For both ModelNet 40 and ShapeNet (see Sec.3.1), we sort the classes by their class ID, and then divide them into four folds. The classes in the first fold are used as testing, and the rest of the classes are used for training. Instead of using all 14 projections generated from a mesh sample, in each experiment, we only use projections belonging to B_i, H_i or T_i during training and testing. Intersection over Union (IoU) of each class is shown in Table 1. As can be seen, for each class, the accuracy varies considerably depending on whether the support images come from B_i, H_i or T_i, and the standard deviation of IoU value becomes as high as 12.18% and 9.33% for ModelNet40 and ShapeNet datasets, respectively. Motivated by these observations, we propose AINet (described in Sec. 4.2, which infers a query image's view angle, and then gives more weight to the support projections, taken under similar view angles, during the testing phase.

Class Name	B_i	H_i	T_i	**Stdev**	**Class Name**	B_i	H_i	T_i	**Stdev**
Airplane	**75.74%**	64.89%	67.86%	5.61%	Bottle	58.14%	**64.51%**	54.55%	5.04%
Bathtub	22.41%	**27.15%**	23.83%	2.43%	Bowl	**50.01%**	38.93%	25.68%	12.18%
Bed	30.31%	**30.87%**	30.66%	0.28%	Car	37.10%	**45.59%**	38.06%	4.65%
Bench	30.45%	**37.44%**	27.74%	5.01%	Chair	**49.43%**	46.57%	44.95%	2.27%
Bookshelf	38.41%	45.75%	**46.69%**	4.53%	Cone	46.39%	**51.35%**	36.20%	7.72%

(a)

Class Name	B_i	H_i	T_i	**Stdev**	**Class Name**	B_i	H_i	T_i	**Stdev**
Airplane	**62.80%**	59.31%	60.35%	1.79%	Birdhouse	**14.92%**	14.22%	12.78%	1.09%
Trash Bin	**38.18%**	37.02%	34.28%	2.00%	Bookshelf	23.29%	21.72%	**25.11%**	1.69%
Bag	**16.33%**	15.80%	16.32%	0.31%	Bottle	51.98%	**53.90%**	51.73%	1.19%
Basket	15.53%	14.81%	**16.83%**	1.02%	Bowl	51.78%	50.06%	34.83%	9.33%
Bathtub	**22.38%**	19.53%	18.10%	2.18%	Bus	**47.45%**	31.14%	40.69%	8.20%
Bed	24.50%	**28.68%**	23.02%	2.94%	Cabinet	23.56%	**30.59%**	27.73%	3.54%
Bench	**32.76%**	28.43%	27.68%	2.74%	Camera	**17.35%**	14.93%	15.06%	1.36%

(b)

Table 1. (a) and (b) show the IoU value for different classes, for a 5-way 1-shot 10 query experiment on ModelNet40 and ShapeNet datasets, respectively. Only B_i (Bottom), H_i (Horizontal) or T_i (Top) views are used as support images. Stdev is the standard deviation of IoU for each class.

4. Proposed Method

We first explain our pretraining method in Sec. 4.1. For fair comparison, the same well-trained backbone is used for all models in all the experiments. We provide the details of our proposed Angle Inference Network (AINet) in Sec. 4.2.

4.1. Pretraining

In fully-supervised learning, a backbone is usually pretrained by using the ImageNet [6], which is a very large dataset containing objects in different color, scale, orientation etc. for each class, so that backbone sees many different variations of objects. In contrast, with few-shot learning, a network should be able to identify the similarities

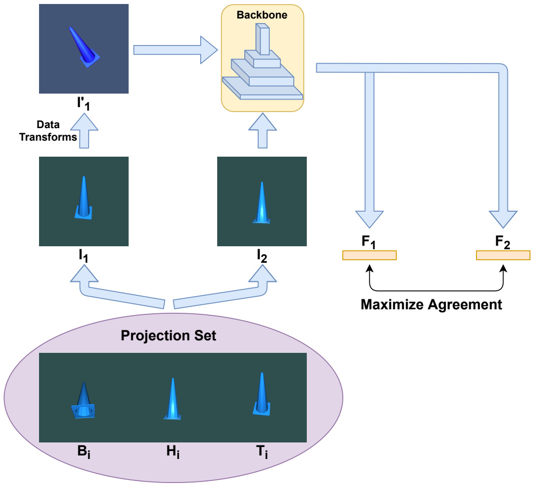

Figure 4. Flow chart of the pretraining of a ResNet backbone.

between support and query images belonging to the same never-before-seen class, instead of learning features of specific objects from different classes. Pretraining a backbone on a large dataset is conflicting for the evaluation of a few-shot model, since class(es) used for testing might have been seen by the backbone during pretraining, and this might result in a misleading, high performance.

Instead, to make the backbone robust to variations in rotation, scale, color, etc., we adopt the strategy in [4] for pretraining. Fig. 4 shows the flow of pretraining, which is performed only on the support projection images obtained from the 3D mesh data of training classes. ResNet is used as the backbone. Since the view angle of each projection is known, we pick a set of $\{B_i, H_i, T_i\}$ for the sane $i \in \{1, 2, 3, 4\}$. As shown in Fig. 2, $\{B_i, H_i, T_i\}$ are the projections taken from the same side of an object, thus sharing some similarities. Then, two images I_1 and I_2 are randomly picked from $\{B_i, H_i, T_i\}$. A data transformation is applied to I_1, and the transformed image I_1' is fed into the backbone together with I_2. Then, feature vectors F_1 and F_2, corresponding to inputs I_1' and I_2 are obtained, and the agreement between these feature vectors are maximized. Please refer to [4] for more details. Since I_1' and I_2 correspond to images with different color, rotation etc., and agreement between them is maximized, the well-pretrained backbone should be robust to image variances, and has initial ability to find similarities between two images. In Sec. 5, when different few-shot heads are tested, the same well-trained backbone is used for fair comparison.

4.2. Angle Inference Network

In our proposed 3DG2D approach, 14 projection images taken under different angles have a great variance, and a query image may be similar to only some of these projections. Thus, treating these different-looking support images equally, which is a common practice in most existing approaches [10, 11, 13, 15, 21, 23], may cause issues. To address this, and motivated by our findings in Sec. 3.2, we propose the Angel Inference Network (AINet), whose structure is shown in Fig. 5. If a query image's view angle is known, model can put more attention on projections sharing the similar view angle as the query image, and achieve better performance. Yet, during both training and testing, query images' angle is unknown. AINet infers the view angle of a query image first.

We first divide 14 projections into three angle sets (Bottom, Horizontal, Top) based on a projection's view angle. We introduce an angle inference loss, and during training, we not only minimize the image classification loss, but also the angle inference loss. When AINet performs prediction during testing, instead of measuring the similarity between the support and query samples directly, the view angle of a query image is first estimated by the marginal distribution:

$$P(A) = \sum_C P(A, C) = \sum_C P(A|C) * P(C), \tag{1}$$

where A and C denote angle category and image class of a query image, respectively. According to Eq. (1), to infer the view angle distribution, $P(A)$, a loss function needs to be designed to optimize $P(C)$ and $P(A|C)$.

Optimization of $P(C)$: As shown in Fig. 5 (a), support projections and query images are first fed into the shared ResNet backbone to obtain their features. Then, we take the average of all the projection features belonging to class j, to obtain the class prototype S_j. Then, the classification loss L_C is calculated as shown on the left of Fig. 5(b). The distance vector $V \in \mathbb{R}^N$ is obtained by calculating the Euclidean distance between each class prototype S_i and the query feature, Where N is the number of ways/classes. If the prototype and query come from the same class, the corresponding distance is minimized by using the classification loss L_C, which is the cross-entropy loss.

Optimization of $P(A|C)$: Since a query image's view angle is not known during training, $P(A|C)$ cannot be optimized directly. Instead, we have

$$P(A|C) = \frac{P(C|A) * P(A)}{P(C)}. \tag{2}$$

Thus $P(A|C) \propto P(C|A)$, and instead of optimizing $P(A|C)$ directly, we can optimize $P(C|A)$, e.g. by letting $P(C = target|A) > P(C \neq target|A)$, for each angle category of bottom (B), horizontal (H) and top (T). The details of the optimization of $P(C|A)$ are shown on the right block of Fig. 5 (b). For each class, the angle prototypes Pr_H, Pr_B and Pr_T are obtained by averaging the projection features belonging to each angle category of H,B and T, respectively. The distance matrix $D \in \mathbb{R}^{N\times 3}$ is obtained by calculating the Euclidean distance between each angle prototype and query feature, where N is the number of

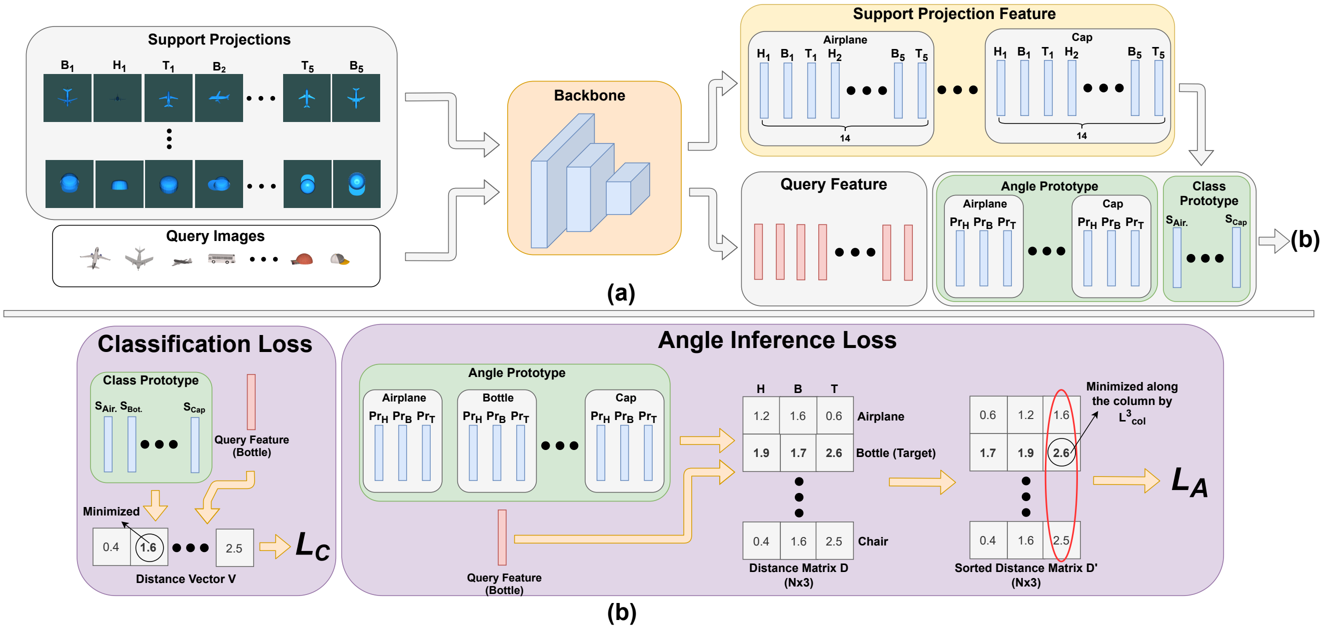

Figure 5. (a) The network architecture. Support projections and query images are fed into the ResNet backbone to obtain their features; (b) In the left block, class j's prototype S_j is obtained by averaging features of all projections belonging to class j. Classification loss L_C is the cross entropy loss. In the right block, angle prototype Pr_i is obtained by averaging the features of the projections in the same angle group (B,T, or H). Pr_i is used to compute the angle inference loss L_A.

classes/ways. To optimize $P(C|A)$, we could directly minimize the target distance in each column. However, there may be cases, for which this distance could be smaller for non-target class projections compared to target class projections for a specific angle category. If the distance in each column of D is minimized directly, the network may get confused. To address this problem, we sort each row of D, and obtain D'. We compute L^i_{col} by applying the cross-entropy loss to minimize the target distance of i^{th} column of D'. Then, the angle inference loss is computed as $L_A = \sum_i L^i_{col}$. This way, when the target distance is minimized in each column of D', instead of finding the best match for a specific angle, the network is allowed to pick the best object based on the best angle category. During training, the total loss is

$$L = (1 - \alpha) * L_C + \alpha * L_A, \tag{3}$$

wherein $\alpha \in [0, 1]$ is a hyper-parameter.

Angle Inference during Testing: During testing, we first obtain $P(C)$, and then calculate $P(A|C)$ by applying Softmax function along each row of the distance matrix D. Once we have $P(C)$ and $P(A|C)$, $P(A) \in \mathbb{R}^3$ can be estimated by Eq. (1). Now that we have an estimate for P(A), we can better estimate the prototype S'_j of each class by incorporating the angle bias as:

$$S'_j = \sum_{i=1}^{3} Pr_i * P_i(A), \tag{4}$$

where Pr_i is the angle prototype shown in Fig 5 (b), and $P_i(A)$ is the probability that the query was taken under i^{th} angle category. We replace the original prototype feature S with S'_j in the left block of Fig. 5(b), and the final prediction result $P'(C)$ is obtained the same way as obtaining $P(C)$.

5. Experiments

We perform experiments on the ModelNet40 [22], Toys4K [14] and ShapeNet [3] datasets. To make it more challenging and realistic, when choosing the query and support images, we make sure that the 3D mesh data that is used to generate the projections come from different object samples in the same class. In addition, we use the projections generated by [14] as query images, which have different distributions of color, texture, and view angle compared to the projections we use in the support set. We compare our proposed AINet with six baseline methods. It should be noted that all the baseline methods were proposed for traditional 2D image few-shot learning (Fig. 1 a), and not for using the 3D projection images as support. In our comparison, we first build the support sets from the projection images, and then apply these baselines. For commensurate comparison, the same well-trained ResNet-10 [9] is used as the backbone for all the methods being compared.

5.1. Few-shot Image Classification on ModelNet40

ModelNet40 dataset contains 12,311 3D objects from 40 classes. We sort the classes alphabetically based on the first letter of the class name, and then split them into four folds, with 10 classes in each fold. At each experiment fold, we pick the 10 classes in one of these folds as testing classes, and use the remaining 30 classes for training. We repeat this four times for 4-fold cross validation. We perform 5-way, 1-shot, 10-query and 5-way, 3-shot, 10-query experiments. Here n-shot refers to using n-many 3D meshes, and thus $14n$ projections, for each class. The accuracy values with the 95% confidence interval are shown in Tab. 2. In 1-shot classification task, our proposed AINet outperforms all other baselines for each fold. In 3-shot classification task, our AINet achieves the best performance for three of the four folds. As for the Mean Accuracy, AINet outperforms the second best model by 1.48% and 0.54%, in 1-shot and 3-shot experiments, respectively. ProtoNet [13], an earlier work, has the simplest architecture, and provides the 2nd or 3rd best performance when compared with more recent works [11, 21, 23].

In our proposed 3DG2D few-shot image classification approach, 14 projections are generated from each 3D mesh shot, and these 14 projections in the support set show significant variance among themselves, making the problem more challenging for the traditional baselines, which pick the 2D support and query images from the same image pool.

	Fold 0	Fold 1	Fold 2	Fold 3	Mean
MetaOpt [10]	68.87%±0.87%	57.13%±0.90%	50.44%±0.83%	54.62%±0.84%	57.77%±0.86%
ADM [11]	65%±0.90%	58.27%±0.86%	47.28%±0.83%	51.81%±0.86%	55.59%±0.86%
DeepBDC [23]	66.84%±0.86%	60.75%±0.83%	52.19%±0.81%	56.55%±0.84%	59.08%±0.84%
RelationNet [15]	64.82%±0.89%	63.77%±0.82%	50.01%±0.84%	54.93%±0.87%	58.39%±0.86%
FRN [21]	71.43%±0.87%	59.9%±0.78%	53.16%±0.92%	58.9%±0.83%	60.85%±0.85%
ProtoNet [13]	72.04%±0.85%	68.53±%0.75%	55.71%±0.84%	59.18%±0.86%	63.86%±0.83%
AINet (ours)	**73.47%±0.79%**	**69.56%±0.76%**	**56.59%±0.78%**	**61.77%±0.80%**	**65.34%±0.78%**

(a) 1-3D (14 projections)-shot image classification result

	Fold 0	Fold 1	Fold 2	Fold 3	Mean
MetaOpt [10]	74.53%±0.67%	62.05%±0.79%	55.17%±0.82%	62.36%±0.77%	63.53%±0.76%
ADM [11]	78.72%±0.67%	61.65%±0.72%	55.64%±0.76%	62.27%±0.76%	64.57%±0.73%
DeepBDC [23]	77.29%±0.65%	73.37%±0.62%	61.11%±0.76%	69.71%±0.67%	70.37%±0.68%
RelationNet [15]	71.5%±0.72%	69.21%±0.70%	56.11%±0.79%	65.59%±0.70%	65.61%±0.73%
FRN [21]	81.53%±0.67%	76.15%±0.63%	62.67%±0.85%	**70.31%±0.68%**	72.67%±0.71%
ProtoNet [13]	81.94%±0.60%	76.05%±0.61%	65.26%±0.75%	65.59%±0.76%	72.21%±0.68%
AINet (ours)	**83.18%±0.58%**	**76.43%±0.63%**	**65.29%±0.75%**	67.94%±0.73%	**73.21%±0.67%**

(b) 3-3D (42 projections)-shot image classification result

Table 2. Accuracy values on the ModelNet40 dataset

5.2. Few-shot Image Classification on Toys4K

Toys4K dataset consists of 4179 instances from 105 categories. Similar to the experiment with ModelNet40, we split the classes into four folds, with 25,25,25, and 30 classes in each fold. We perform 4-fold cross validation. The results are shown in Tab. 3. Our proposed AINet achieves the best performance for all folds in the 1-shot classification task, and for 3 of the 4 folds in the 3-shot classification. As for the mean accuracy, AINet outperforms the second best model, ProtoNet, by 1.64% and 0.32% on 1-shot and 3-shot classification tasks, respectively.

	Fold 0	Fold 1	Fold 2	Fold 3	Mean
MetaOpt	56.7%±0.90%	58.19%±1.0%	65.58%±0.96%	56.38%±0.98%	59.21%±0.96%
ADM	55.58%±0.87%	55.74%±0.99%	64.69%±0.96%	55.14%±0.99%	57.79%±0.95%
DeepBDC	58.53%±0.97%	60.05%±0.95%	66.51%±0.89%	61.42%±0.96%	61.63%±0.94%
RelationNet	60.33%±0.91%	61.87%±0.95%	65.89%±0.98%	62.38%±0.99%	62.62%±0.96%
FRN	59.11%±0.98%	58.45%±1.0%	65.97%±1.0%	60.43%±0.98%	60.99%±0.99%
ProtoNet	62.96%±0.91%	62.51%±0.92%	69.18%±0.96%	62.76%±0.93%	64.35%±0.93%
AINet (ours)	**63.86%0.96%**	**63.95%±0.91%**	**71.73%±0.93%**	**64.41%±0.94%**	**65.99%±0.94%**

(a) 1-3D (14 projections)-shot image classification result

	Fold 0	Fold 1	Fold 2	Fold 3	Mean
MetaOpt	62.48%±0.93%	66.09%±0.92%	72.48%±0.85%	63.69%±0.89%	66.19%±0.90%
ADM	61.24%±0.88%	61.55%±0.92%	65.5%±0.88%	62.14%±0.86%	62.61%±0.89%
DeepBDC	68.96%±0.77%	71.09%±0.88%	74.39%±0.79%	67.63%±0.85%	70.52%±0.82%
RelationNet	69.18%±0.78%	70.64%0.79%	73.11%±0.90%	70.37%±0.83%	70.83%±0.83%
FRN	69.94%±0.82%	69.07%±0.88%	76.57%±0.81%	**70.89%±0.83%**	71.62%±0.84%
ProtoNet	72.11%±0.79%	71.63%±0.85%	77.99%±0.77%	70.1%±0.85%	72.96%±0.82%
AINet (ours)	**72.6%±0.77%**	**71.92%±0.80%**	**78.55%±0.77%**	70.06%±0.84%	**73.28%±0.80%**

(b) 3-3D (42 projections)-shot image classification result

Table 3. Accuracy values on the Toys4K dataset

5.3. Few-shot Image Classification on ShapeNet

ShapeNet contains 52K mesh samples from 55 object categories with basic surface texture properties. We sort 55 classes by their class ID in an ascending order, and split them into four folds, with 14,14,14 and 13 classes in each fold. We perform 4-fold cross validation. The results are shown in Tab, 4. Our propose AINet provides the best performance for all folds in 1-shot classification task and for 3 of the 4 folds in 3-shot classification. As for the mean accuracy, our proposed AINet outperforms the second best model by 0.82% and 0.64% in 1-shot and 3-shot classification task, respectively.

	Fold 0	Fold 1	Fold 2	Fold 3	Mean
MetaOpt	53.06%±0.92%	64.44%±0.90%	58.8%±0.80%	54.67%±0.90%	57.74%±0.88%
ADM	52.88%±0.91%	62.28%±0.90%	54.48%±0.88%	57.74%±0.81%	56.85%±0.88%
DeepBDC	53.91%±0.93%	65.8%±0.89%	56.73%±0.82%	59.13%±0.86%	58.89%±87.5%
RelationNet	53.4%±0.91%	65.57%±0.89%	58.03%±0.81%	58.86%±0.82%	58.97%±85.75%
FRN	55.93%±0.91%	68.48%±0.91%	58.99%±0.87%	59.12%±0.85%	60.63%±88.5%
ProtoNet	58.18%±0.93%	69.53%±0.86%	61.79%±0.83%	60.99%±0.82%	62.62%±0.86%
AINet (ours)	**58.9%±0.89%**	**70.5%±0.89%**	**62.86%±0.84%**	**61.51%±0.81%**	**63.44%±0.86%**

(a) 1-3D (14 projections)-shot image classification result

	Fold 0	Fold 1	Fold 2	Fold 3	Mean
MetaOpt	59.96%±0.89%	70.4%±0.85%	64.53%±0.80%	59.73%±0.80%	63.66%±0.84%
ADM	56.54%±0.84%	71.19%±0.85%	65.71%±0.77%	67.4%±0.72%	65.21%±0.80%
DeepBDC	65.56%±0.81%	77.69%±0.70%	**70.69%±0.75%**	68.09%±0.72%	70.51%±0.75%
RelationNet	61.59%±0.83%	72.39%±0.75%	66.02%±0.84%	64.63%±0.74%	66.16%±0.79%
FRN	63.75%±0.86%	76.2%±0.81%	68.05%±0.84%	66.89%±0.73%	68.72%±0.81%
ProtoNet	66.53%±0.81%	77.57%±0.71%	69.92%±0.75%	69.08%±0.69%	70.78%±0.74%
AINet (ours)	**67.13%±0.85%**	**78.92%±0.69%**	70.09%±0.71%	**69.56%±0.71%**	**71.42%±0.74%**

(b) 3-3D (42 projections)-shot image classification result

Table 4. Accuracy values on the ShapeNet dataset

5.4. Experiments using RGB Images as Query

We also performed experiments to show that our method can be applied when RGB images are used as queries, i.e. instead of using the projection images, generated by [14] as queries, we use RGB images collected from Web, which are split into set (a) and set (b). A few examples are shown in Fig. 6. Please see the suppl. material for the rest. We also perform a comparison with the original ProtoNet. Since ProtoNet, and other traditional approaches, randomly pick support and query images from the same RGB image pool, without considering variations in view angle, in our experiments, for each class, an angle category (B, H or T) is picked first. Then, projections from that specific angle category are used as the support set for ProtoNet. In Tab. 5, the IoU value for each class is shown for the experiments performed on query set (a) and (b). Our proposed method performs well on RGB images. As for ProtoNet, since the support set is composed of images from one view angle, the

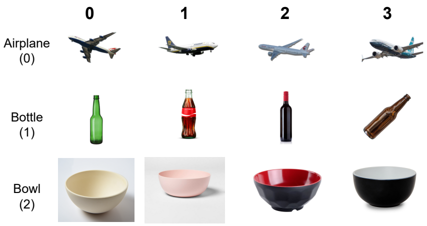

Figure 6. Example query images collected from the Web. 20 images are collected for each class.

	Airplane	Bottle	Bowl	Chair	Cone	Mean
ProtoNet	0.00%	7.14%	20.00%	0.00%	4.76%	6.38%
Ours	100.00%	80.00%	81.82%	81.82%	69.23%	82.57%

(a) Experiment result on query set (a)

	Airplane	Bottle	Bowl	Chair	Cone	Mean
ProtoNet	0.00%	26.32%	7.69%	0.00%	4.76%	7.75%
Ours	90.00%	90.91%	90.00%	90.91%	100.00%	92.36%

(b) Experiment result on query set (b)

Table 5. IoU value for each class for different query sets.

model fails on this task. Please see the Supp. material for details, and additional experiments with RGB images.

6. Ablation Studies

In these studies, we perform comparisons with the traditional approach of finding the class prototype by taking the average of the support features, as done in ProtoNet. We list this as ProtoNet in the tables, even though original ProtoNet does not use 3D data projections as support.

6.1. Analysis of Angle Inference Loss

We first analyze the contribution of L_A, which was introduced in Sec. 4.2, by setting α in Eq. (3) to zero, and thus removing L_A from the loss. By doing so, during training, features of all the support images are averaged, as done in ProtoNet. As for testing, AINet performs angle inference by (1) for final prediction. The results of the experiment, performed on ModelNet40, are shown in Tab. 6. Without the use of L_A, the AINet still outperforms ProtoNet for all folds by inferring the view angle of a query image during testing phase. AINet with L_A achieves the best performance in 7 of the 8 test folds, since the angle inference during testing is not that accurate without the use of L_A.

6.2. Analysis of Hyperparameter α

To study the effect of hyperparameter α on AINet's performance, we set α to values between 0.1 and 0.5. The results of the experiment performed on ModelNet40 dataset for 1-shot image classification task, are shown in Table 7. For different α values, AINet outperforms the traditional approach in 13 out of the 20 experiments (numbers in black font). In terms of mean accuracy, AINet provides better performance for all α values.

	Fold 0	Fold 1	Fold 2	Fold 3	Mean
ProtoNet	72.04%	68.53%	55.71%	59.18%	63.86%
AINet (w/o L_A)	72.28%	69.54%	56.07%	59.69%	64.40%
AINet (with L_A)	**73.47%**	**69.56%**	**56.59%**	**61.77%**	**65.34%**

(a) 1-3D (14 projections)-shot image classification result

	Fold 0	Fold 1	Fold 2	Fold 3	Mean
ProtoNet	81.94%	76.05%	65.26%	65.59%	72.21%
AINet (w/o L_A)	81.98%	76.22%	**65.29%**	65.86%	72.34%
AINet (with. L_A)	**83.18%**	**76.43%**	65.21%	**67.94%**	**73.19%**

(b) 3-3D (42 projections)-shot image classification result

Table 6. Classification accuracy with and w/o the angle inference loss L_A. The best and the 2nd best performances are marked in bold and blue color, respectively.

α	Fold 0	Fold 1	Fold 2	Fold 3	Mean
0.1	72.82%	67.42%	**56.58%**	59.94%	64.19%
0.2	73.25%	69.17%	54.98%	60.61%	64.50%
0.3	**74.46%**	**69.56%**	55.67%	60.94%	**65.16%**
0.4	73.05%	68.18%	55.06%	**61.76%**	64.51%
0.5	73.40%	68.06%	55.45%	60.95%	64.46%

Table 7. Accuracy for different α values. The best performance in each column is shown in bold. If accuracy is lower than the traditional averaging approach, the value is marked in red.

6.3. Analysis of Number of Angle Categories

We analyzed the performance when 14 angle categories, instead of three (Bottom, Horizontal and Top) are used. The results in the Suppl. material show that AINet with the three angle categories achieves the best performance.Moreover, regardless of the number of angle categories, the performance of AINet is higher than ProtoNet.

6.4. Analysis of Pretraining

We provide an analysis of how the pretraining affects the network's performance in the supplementary material.

7. Conclusion

In this paper, we have first presented how having view-angle and shape variety in the support set affects few-shot classification performance. Motivated by our findings, we have proposed a 3D Guided 2D (3DG2D) few-shot image classification approach, wherein projections of 3D mesh data, taken from different view angles, serve as the support set to classify 2D query images. After showing that projections generated from different view angles contribute differently to the classification of query images during testing, we have proposed an Angle Inference Network (AINet). With AINet, more weight is given to support projections sharing similar view angles with query images. We have performed experiments with cross validation on the ModelNet40, Toys4K and ShapeNet datasets, and shown that AINet consistently outperforms the SOTA approaches. We have also performed experiments using RGB images, collected from Web as queries, and presented ablation study results. In future work, experiments with RGB images will be extended by collecting more images covering more classes.

References

[1] Kelsey Allen, Evan Shelhamer, Hanul Shin, and Joshua Tenenbaum. Infinite mixture prototypes for few-shot learning. In *International Conference on Machine Learning*, pages 232–241. PMLR, 2019.

[2] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6, 1993.

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[5] Lina I Davitt, Filipe Cristino, Alan C-N Wong, and E Charles Leek. Shape information mediating basic-and subordinate-level object recognition revealed by analyses of eye movements. *Journal of Experimental Psychology: Human Perception and Performance*, 40(2):451, 2014.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] David H Foster and Stuart J Gilson. Recognizing novel three–dimensional objects by summing signals from parts and views. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 269(1503):1939–1947, 2002.

[8] Jiechao Guan, Zhiwu Lu, Tao Xiang, Aoxue Li, An Zhao, and Ji-Rong Wen. Zero and few shot learning with semantic feature synthesis and competitive learning. *IEEE transactions on pattern analysis and machine intelligence*, 43(7):2510–2523, 2020.

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

[10] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10657–10665, 2019.

[11] Wenbin Li, Lei Wang, Jing Huo, Yinghuan Shi, Yang Gao, and Jiebo Luo. Asymmetric distribution measure for few-shot learning. *arXiv preprint arXiv:2002.00153*, 2020.

[12] Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Few-shot learning with embedded class models and shot-free meta training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 331–339, 2019.

[13] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[14] Stefan Stojanov, Anh Thai, and James M Rehg. Using shape to categorize: Low-shot learning with an explicit shape bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1798–1808, 2021.

[15] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

[16] Gábor J Székely and Maria L Rizzo. Brownian distance covariance. *The annals of applied statistics*, 3(4):1236–1265, 2009.

[17] Gábor J Székely, Maria L Rizzo, and Nail K Bakirov. Measuring and testing dependence by correlation of distances. *The annals of statistics*, 35(6):2769–2794, 2007.

[18] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[19] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7278–7286, 2018.

[20] Davis Wertheimer, Luming Tang, and Bharath Hariharan. Few-shot classification with feature map reconstruction networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8012–8021, 2021.

[21] Davis Wertheimer, Luming Tang, and Bharath Hariharan. Few-shot classification with feature map reconstruction networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8012–8021, June 2021.

[22] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[23] Jiangtao Xie, Fei Long, Jiaming Lv, Qilong Wang, and Peihua Li. Joint distribution matters: Deep brownian distance covariance for few-shot classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7972–7981, 2022.

[24] Jiangtao Xie, Fei Long, Jiaming Lv, Qilong Wang, and Peihua Li. Joint distribution matters: Deep brownian distance covariance for few-shot classification. In *CVPR*, 2022.

[25] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12203–12213, 2020.

[26] Hongguang Zhang, Jing Zhang, and Piotr Koniusz. Few-shot learning via saliency-guided hallucination of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2770–2779, 2019.