Generating Representative Samples for Few-Shot Classification

Jingyi Xu
Stony Brook University
jingyixu@cs.stonybrook.edu

Hieu Le*
Amazon Robotics
ahieu@amazon.com

Abstract

Few-shot learning (FSL) aims to learn new categories with a few visual samples per class. Few-shot class representations are often biased due to data scarcity. To mitigate this issue, we propose to generate visual samples based on semantic embeddings using a conditional variational autoencoder (CVAE) model. We train this CVAE model on base classes and use it to generate features for novel classes. More importantly, we guide this VAE to strictly generate representative samples by removing non-representative samples from the base training set when training the CVAE model. We show that this training scheme enhances the representativeness of the generated samples and therefore, improves the few-shot classification results. Experimental results show that our method improves three FSL baseline methods by substantial margins, achieving state-of-the-art few-shot classification performance on miniImageNet and tieredImageNet datasets for both 1-shot and 5-shot settings. Code is available at: https://github.com/cvlab-stonybrook/fsl-rsvae.

1. Introduction

Few-shot learning (FSL) methods aim to learn useful representations with limited training data. They are extremely useful for situations where machine learning solutions are required but large labelled datasets are not trivial to obtain (e.g. rare medical conditions [49, 71], rare animal species [75], failure cases in autonomous systems [42, 43, 58]). Generally, FSL methods learn knowledge from a fixed set of base classes with a surplus of labelled data and then adapt the learned model to a set of novel classes for which only a few training examples are available [73].

Many FSL methods [10, 23, 39, 65, 65, 77, 82] employ a prototype-based classifier for its simplicity and good performance. They aim to find a prototype for each novel class such that it is close to the testing samples of the same class and far away from testing samples for other classes. However, it is challenging to estimate a representative prototype just from a few available support samples [37, 79]. An effective strategy to enhance the representativeness of the prototype is to employ textual semantic embeddings learned via NLP models [13, 46, 52, 53] using large unsupervised text corpora [77, 82]. These semantic embeddings implicitly associate a class name, such as “Yorkshire Terriers”, with the class representative semantic attributes such as “smallest dog” or “long coat” [1] (Fig. 1), providing strong and unbiased priors for category recognition.

For the most part, current FSL methods focus on learning to adaptively leverage the semantic information to complete the original biased prototype estimated from the few available samples. For example, the recent FSL method of Zhang et al. [82] learns to fuse the primitive knowledge and attribute features into a representative prototype, depending on the set of given few-shot samples. Similarly, Xing et al. [77] propose a method that computes an adaptive mixture coefficient to combine features from the visual and tex-
tual modalities. However, learning to recover an arbitrarily biased prototype is challenging due to the drastic variety of the possible combinations of few-shot samples.

In this paper, we propose a novel FSL method to obtain class-representative prototypes. Inspired by zero-shot learning (ZSL) methods [4, 18, 85], we propose to generate visual features via a variational autoencoder (VAE) model [66] conditioned on the semantic embedding of each class. This VAE model learns to associate a distribution of features to a conditioned semantic code. We assume that such association generalizes across the base and novel classes [3, 47]. Therefore, the model trained with sufficient data from the base classes can generate novel-class features that align with the real unseen features. We then use the generated features together with the few-shot samples to construct class prototypes. We show that this strategy achieves state-of-the-art results on both MiniImageNet and tieredImageNet datasets. It works exceptionally well for 1-shot scenarios where our method outperforms state-of-the-art methods [76, 80] by 5 ∼ 6% in terms of classification accuracy.

Moreover, to enhance the representativeness of the prototype, we guide the VAE to generate more representative samples. Here we refer representative samples to the "easy-to-recognize" samples that faithfully reflect the key characteristics of the category (see Fig. 1). The embeddings of these representative samples often lie close to their corresponding class centers, which are particularly useful for constructing class-representative prototypes.

Specifically, we guide the VAE model to generate representative samples by selecting only representative data from the base classes for training it. In essence, our VAE model is trained to model the data distribution of the training set. As the training set contains only representative data, the trained VAE model outputs samples that are also representative. Specifically, to select those representative features, we first assume that the feature vectors of each class follow a multivariate Gaussian distribution and estimate this distribution for each base class. Based on these distributions, we compute the probability of each sample belonging to its corresponding category to measure the representativeness for the sample. We filter out the non-representative samples and train the VAE using only representative samples. Interestingly, we show that the representativeness of the training set highly corresponds to the accuracy of the few-shot classifier. We obtain the highest accuracy when training the VAE with the most representative samples. In this case, we only use a small percentage of the whole training set, e.g., 10% for the case of MiniImageNet dataset, to obtain the best results. Our analyses show that this approach consistently improves the FSL classification performance by 1 ∼ 2% across all benchmarks for three different baselines [10, 39, 65].

Our main contributions can be summarized as follows:

• We are the first to use a VAE-based feature generation approach conditioned on class semantic embeddings for few-shot classification.

• We propose a novel sample selection method to collect representative samples. We use these samples to train a VAE model to obtain reliable data points for constructing class-representative prototypes.

• Our experiments show that our methods achieve state-of-the-art performance on two challenging datasets, tieredImageNet and miniImageNet.

We summarize related FSL works in Section 2. Section 3 provides a rundown of our approach. Section 4 reports the main results obtained with our method. In section 5, we provide multiple analyses to clarify different aspects of our methods.

2. Related Work

Few-shot Learning. FSL is helpful when we only have limited labeled training data [7, 25–30]. Representative FSL approaches include metric learning based [65, 67, 68, 70, 79, 80, 83], optimization based [17, 31, 33, 34, 37, 54, 59, 62], and data augmentation based methods [2, 61, 74, 78]. Similar to our method, some FSL methods use semantic information to improve the few-shot classifiers [21, 51, 69, 77, 82]. Zhang et al. [82] and Xing et al. [77] propose methods that learn to adaptively combine the visual features and the semantic features to obtain an unified cross-modality representation for each class. These two methods focus on the fusing strategies that combine features of the two domains. Hu et al. [21] propose to disentangle the visual features into the sub-spaces that associate to different semantic attributes. The FSL method of Peng et al. [51] uses semantic information to infer a classifier for novel classes and adaptively combines this classifier with the few-shot samples. Our method is the first FSL method that uses a conditional VAE model to directly generate visual features, conditioned on the semantic embedding of each class.

Conditional Variational Autoencoder. The practice of using a conditional VAE to model a feature distribution has been used before in many computer vision tasks such as image classification [23, 60, 78, 84], image generation [16, 38], image restoration [14], or video processing [50]. Using VAE models for generating features conditioned on the corresponding semantic embedding is fairly common in ZSL methods [4, 18, 47, 60, 81, 85]. Mishra et al. [47] are the first to propose to use a conditional VAE for ZSL where they view ZSL as a case of missing data. They find that such an approach can handle well the domain shift problem. Similarly, Arora et al. [3] show that a conditional VAE can
be used together with a GAN system to synthesize images for unseen classes effectively. Keshari et al. [22] focus on generating a specific set of hard samples which are closer to another class and the decision boundary. For the most part, ZSL methods aim to model the whole distribution of data [6,9,40,60], while our method focuses on modeling the distribution of representative samples useful for constructing the class-representative prototypes.

**Sample Selection.** To the best of our knowledge, we are the first to propose using a sample selection method for selecting training samples for a VAE model. Here we select only representative samples for training the VAE. This is a new sample selection regime since mainstream sample selection works mainly focus on identifying the most informative samples [5, 24] for training their models, which is widely used in active-learning [32, 63]. In FSL, Chang et al. [8] propose a method to select the most informative data that should be annotated for a few-shot text generation system. Zhou et al. [86] propose a method to select the useful base classes to train their model, while our work selects useful individual samples within an arbitrary set of base classes.

### 3. Method

#### 3.1. Problem Definition

In a typical few-shot classification setting, we are given a set of data-label pairs $D = \{(x^i, y^i)\}$. Here $x^i \in \mathbb{R}^d$ is the feature vector of a sample and $y^i \in C$, where $C$ denotes the set of classes. The set of classes is divided into base classes $C_b$ and novel classes $C_n$. The sets of class $C_b$ and $C_n$ are disjoint, i.e., $C_b \cap C_n = \emptyset$. For a $N$-way $K$-shot problem, we sample $N$ classes from the novel set $C_n$, and $K$ samples are available for each class. $K$ is often small (i.e., $K = 1$ or $K = 5$). Our goal is to classify query samples correctly using the few samples from the support set.

### 3.2. Overall Pipeline

Fig. 2 gives an overview of our sample selection method and VAE training approach. We propose a method to select a set of representative samples from a set of base classes. We use these selected representative data to train a conditional VAE model for feature generation. To select representative samples, we assume that the features of each class follow a multivariate Gaussian distribution. We estimate the parameters for each class distribution and compute the probability for each data point belonging to its class. By setting a threshold on the probabilities, we identify a set of representative samples. We then use these selected representative samples to train a VAE model that generates samples conditioned on the semantic attributes of each class.

We train this VAE on the base classes and use the trained model to generate samples for the novel classes. The generated features are then used together with the few-shot samples to construct the prototype for each class. Our method is a simple plug-and-play module and can be built on top of any pretrained feature extractors. In our experiments, we show that our method consistently improves three baseline few-shot classification methods: Meta-Baseline [10], ProtoNet [65] and E3BM [39] by large margins.
3.2.1 Class-representative Sample Selection

In this paper, we are interested in representative samples as they can serve as reliable data points for constructing a class-representative prototype \[10, 65\]. The main idea is to train a feature generator with only representative data to obtain more representative generated samples.

To select the representative features, we assume that the feature distribution of the base classes follows a Gaussian distribution and estimate the parameters of this distribution for each class. We calculate the Gaussian mean of a base class \(i\) as the mean of every single dimension in the vector:

\[
\mu_i = \frac{1}{n_i} \sum_{j=1}^{n_i} x^j, \tag{1}
\]

where \(x^j\) is a feature vector of the \(j\)-th sample from the base class \(i\) and \(n_i\) is the total number of samples in class \(i\). The covariance matrix \(\Sigma_i\) for the distribution of class \(i\) is calculated as:

\[
\Sigma_i = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (x^j - \mu_i)(x^j - \mu_i)^T. \tag{2}
\]

Once we estimate the parameters of the Gaussian distribution using the adequate samples from the base classes, the probability density of observing a single feature, \(x^j\), being generated from the Gaussian distribution of class \(i\) is given by:

\[
p(x^j | \mu_i, \Sigma_i) = \frac{\exp\left\{ -\frac{1}{2} (x^j - \mu_i)^T \Sigma_i^{-1} (x^j - \mu_i) \right\}}{(2\pi)^{k/2}|\Sigma_i|^{1/2}}, \tag{3}
\]

where \(k\) is the dimension of the feature vector.

Here we assume that the probability of a single sample belongs to its category’s distribution reflects the representativeness of the sample, i.e., the higher the probability, the more representative the sample is. By setting a threshold \(\epsilon\) on the estimated probability, we filter out those samples with small probabilities and get a set of representative features for class \(i\):

\[
\mathbb{D}^i = \{ x^j \mid p(x^j | \mu_i, \Sigma_i) > \epsilon \}, \tag{4}
\]

where \(\mathbb{D}^i\) stores the features for class \(i\) with the probabilities larger than a threshold \(\epsilon\).

3.2.2 Conditional VAE Model for Feature Generation

We use our sample selection method to select a set of representative samples and use them for training our feature generation model. We develop our feature generator based on a conditional variational autoencoder (VAE) architecture \[66\] (see Fig. 2b). The VAE is composed of an Encoder \(E(x, a)\), which maps a visual feature \(x\) to a latent code \(z\), and a decoder \(G(z, a)\) which reconstructs \(x\) from \(z\). Both \(E\) and \(G\) are conditioned on the semantic embedding \(a\). The loss function for training the VAE for a feature \(x^j\) of class \(i\) can be defined as:

\[
L_V(x^j) = \text{KL} \left( q(z|x^j, a^i) || p(z|a^i) \right) - \log p(x^j | z, a^i), \tag{5}
\]

where \(a^i\) is the semantic embedding of class \(i\). The first term is the Kullback-Leibler divergence between the VAE posterior \(q(z|x, a)\) and a prior distribution \(p(z|a)\). The second term is the decoder’s reconstruction error. \(q(z|x, a)\) is modeled as \(E(x, a)\) and \(p(x|z, a)\) is equal to \(G(z, a)\). The prior distribution is assumed to be \(\mathcal{N}(0, I)\) for all classes.

The loss for training the feature generator is the loss over all selected representative training samples:

\[
L_V = \sum_{i=1}^{C_b} \sum_{x^j \in \mathbb{D}^i} L_V(x^j) \tag{6}
\]

3.2.3 Constructing Class Prototypes

After the VAE is trained on the base set, we generate a set of features for a class \(y\) by inputting the respective semantic vector \(a^y\) and a noise vector \(z\) to the decoder \(G\):

\[
G^y = \{ \hat{x} | \hat{x} = G(z, a^y), z \sim \mathcal{N}(0, I) \}. \tag{7}
\]

The generated features along with the original support set features for a few-shot task is then served as the training data for a task-specific classifier. Following our baseline methods, we compute the prototype for each class and apply the nearest neighbour classifier. Specifically, we first compute two separated prototypes: one using the support features and the other using the generated features. Each prototype is the mean vector of the features of each group. We then take a weighted sum of the two prototypes to obtain the final prototype \(p^y\) for class \(y\):

\[
p^y = w_g * \frac{1}{|G^y|} \sum_{x^j \in G^y} \hat{x}^j + w_s * \frac{1}{|S^y|} \sum_{x^j \in S^y} x^j, \tag{8}
\]

where \(S^y\) is the support set features and \((w_g, w_s)\) are the coefficients of the generated feature prototype and the real feature prototype, respectively. We classify samples by finding the nearest class prototype for an embedding query feature. We conduct further analysis to show that our generated features can benefit all types of classifiers (see Section 5.2). Compared to the methods that correct the original biased prototype, our model does not require any carefully designed combination scheme.
### 4. Experiments

#### 4.1. Experimental Settings

**Datasets.** We evaluate our method on two widely-used benchmarks for few-shot learning, miniImageNet [55] and tieredImageNet [57]. miniImageNet is a subset of the ILSVRC-12 dataset [12]. It contains 100 classes and each class consists of 600 images. The size of each image is 84 × 84. Following the evaluation protocol of [56], we split the 100 classes into 64 base classes, 16 validation classes, and 8 novel classes for training, validation, and testing.

tieredImageNet is a larger subset of ILSVRC-12 dataset, which contains 608 classes sampled from hierarchical category structure. The average number of images in each class is 1281. It is first partitioned into 34 super-categories that are split into 20 classes for training, 6 classes for validation, and 8 classes for testing. This leads to 351 actual categories for training, 97 for validation, and 160 for testing.

**Baseline methods.** Our method can be used as a simple plug-and-play module for many existing few-shot learning methods without fine-tuning their feature extractors. We investigate three baseline few-shot classification methods used in conjunction with our method: ProtoNet [80], Meta-Baseline [10] and E3BM [39]. ProtoNet is known as a strong and classic prototypical approach. In our experiments, we use the ProtoNet implementation of Ye et al. [80]. Meta-Baseline [10] uses a ProtoNet model to fine-tune a generic classifier via meta-learning. E3BM [39] meta-learns the ensemble of epoch-wise models to achieve robust predictions for FSL. For each baseline method, we extract the corresponding feature representations to train our feature generation VAE model. We then use the trained VAE to generate features and obtain the class prototypes for few-shot classification.

**Evaluation protocol.** We use the top-1 accuracy as the evaluation metric to measure the performance of our method. We report the accuracy on standard 5-way 1-shot and 5-shot settings with 15 query samples per class. We randomly sample 2000 episodes from the test set and report the mean accuracy with the 95% confidence interval.

#### 4.2. Implementation Details

All the three baselines use ResNet12 backbone as the feature extractor. The feature representation is extracted by average pooling the final residual block outputs. The dimension of the feature representation is 640 for ProtoNet [80], 512 for Meta-Baseline [10], and 640 for E3BM [39]. For our feature generation model, both the encoder and the decoder are two-layer fully-connected (FC) networks with 4096 hidden units. LeakyReLU and ReLU [19] are the non-
linear activation functions in the hidden and output layers, respectively. The dimensions of the latent space and the semantic vector are both set to be 512. The network is trained using the Adam optimizer with $10^{-4}$ learning rate. Our semantic embeddings are extracted from CLIP [53]. We empirically set the combination weights $[w_g, w_s]$ in Equation 8 to $[\frac{1}{4}, \frac{1}{2}]$ for 1-shot settings and to $[\frac{1}{6}, \frac{5}{6}]$ for 5-shot settings. We set the probability threshold to 0.9 for the main experiments and discuss the performance under different values of this threshold in Section 5.1.

4.3. Results

Table 1 presents the 5-way 1-shot and 5-way 5-shot classification results of our methods on miniImageNet and tieredImageNet in comparison with previous FSL methods. Here all methods use ResNet12/ResNet18 architectures as feature extractors with input images of size $84 \times 84$. Thus, the comparison is fair. For the rest of the paper, we denote our VAE trained with all data as SV AE, the combination weights as \[ [w_g, w_s] \] in Equation 8 to \[ [\frac{1}{4}, \frac{1}{2}] \] for 1-shot settings and to \[ [\frac{1}{6}, \frac{5}{6}] \] for 5-shot settings. We set the probability threshold to 0.9 for the main experiments and discuss the performance under different values of this threshold in Section 5.1.

5. Analyses

All the following analyses use the feature extractor from the Meta-Baseline method [10].

5.1. Analysis on the Probability Threshold

In our main setting, we set a threshold of 0.9 on the probabilities to select those class-representative samples as the training data for our VAE model (the higher, the more representative). In this section, we conduct experiments with different threshold values to see how it affects the classifier’s performance. Fig. 3 shows the classification accuracy under different thresholds on miniImageNet and tieredImageNet datasets. As the threshold increases, more non-representative samples are filtered out, resulting in less training data for R-SVAE. Interestingly, we observe that the model generally performs better with higher threshold val-
Figure 4. Feature Visualization. We show the t-SNE visualization of the original features (marked as dark points) and our generated features (marked as transparent points) on tiered/ImageNet dataset. Different colors represent different classes. From left to right, we show the original support set (a), the query set (b), the features generated by SVAE (c), and the features generated by R-SVAE (d).

Table 2. 1-shot performance on miniImageNet.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Baseline</th>
<th>SVAE</th>
<th>R-SVAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>95%</td>
<td>96%</td>
<td>97%</td>
</tr>
</tbody>
</table>

Figure 5. Distance Distributions. Kernel Density Estimation of the distance between the estimated prototypes and the ground truth prototype. A smaller value means the estimated prototypes are closer to the ground truth prototypes.

In our main experiments, we classify samples by finding the nearest neighbor among class prototypes. In this section, we apply another three different types of classifiers: 1-nearest neighbor classifier (1-N-N), Support Vector Machine (SVM), and Logistic Regression (LR).

Table 2 shows the 1-shot performance of different classifiers using our generated features on miniImageNet and tiered/ImageNet datasets. It shows that the features generated by our VAEs improve the performance of all three classifiers. For example, the 1-shot accuracy on miniImageNet using LR is improved by 8.8% with SVAE and by 10.1% with R-SVAE. The consistent performance improvements show that our generated features can benefit different types of classifiers.

5.3. Feature Distribution Analysis

In Fig. 4, we show the t-SNE representation [41] of different sets of features for three classes from the novel set of tiered/ImageNet dataset. From left to right, we visualize the distribution of the original support set (a), the query set (b), the features generated by SVAE (c), and the features generated by R-SVAE (d). Note that our methods do not rely on the support features to generate features.

Fig. 4(c) and (d) visualize the effect of our sample selection method. Fig. 4(c) visualizes features generated from our method trained with all available data from the base classes, which consist of 1281 images per class on average. In Fig. 4(d), we train the same model with only 484 representative images per class on average. Our model trained with a representative subset of data generates features that lie closer to the real features, showing the effectiveness of our sample selection method.

Moreover, we plot the distance distributions between the estimated prototypes and the ground truth prototypes of each class. Specifically, for each class, we first obtain the ground-truth prototype by taking the mean of all the features of the class. Then we calculate the $L_2$ distance between the ground truth prototype and three different prototypes: 1) Baseline: the prototype was estimated using only the support samples. 2) SVAE: the prototype was estimated using the support samples and the generated samples from our SVAE model. 3) R-SVAE: the prototype was estimated using the support samples and the generated samples from our R-SVAE model.

We sample 2400 tasks from miniImageNet dataset under both 5-way 1-shot and 5-way 5-shot settings. For each task, we obtain five distances, one distance per class. Then we plot the probability density distribution of the distance, shown in Fig. 5. The probability density is calculated by binning and counting observations and then smoothing them with a Gaussian kernel, namely, Kernel Density Esti-
Figure 6. Examples of representative samples (left) and non-representative samples (right). We visualize 5 images with high probabilities and 5 images with small probabilities computed via our proposed method for 3 classes from tiered ImageNet dataset.

Table 2. Choices of the classifiers. One-shot classification accuracy on miniImageNet and tieredImageNet using different types of classifiers, i.e., 1-N-N, SVM and LR. All methods use the feature extractor from the Meta-Baseline method [10].

<table>
<thead>
<tr>
<th>Classifier</th>
<th>miniImageNet</th>
<th>tieredImageNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prototye [10]</td>
<td>63.17 ± 0.23</td>
<td>72.79 ± 0.19</td>
</tr>
<tr>
<td>1-N-N</td>
<td>63.28 ± 0.23</td>
<td>69.27 ± 0.19</td>
</tr>
<tr>
<td>SVM</td>
<td>63.41 ± 0.23</td>
<td>72.84 ± 0.19</td>
</tr>
<tr>
<td>LR</td>
<td>63.33 ± 0.22</td>
<td>73.41 ± 0.19</td>
</tr>
</tbody>
</table>

Table 3. Classification accuracy using Word2Vec [45] as the semantic feature extractor.

<table>
<thead>
<tr>
<th></th>
<th>1-shot</th>
<th>5-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meta-Baseline</td>
<td>63.17 ± 0.23</td>
<td>79.76 ± 0.17</td>
</tr>
<tr>
<td>Meta-Baseline + SVAE</td>
<td>67.39 ± 0.21</td>
<td>79.77 ± 0.17</td>
</tr>
<tr>
<td>Meta-Baseline + R-SVAE</td>
<td>68.03 ± 0.22</td>
<td>79.93 ± 0.16</td>
</tr>
</tbody>
</table>

5.4. Sample Visualization

In Fig. 6, we visualize some representative samples and non-representative samples based on the representativeness probability computed via our method. The samples on the left panel are images with high probabilities. These images mostly contain the main object of the category and are easy to recognize. On the contrary, the samples on the right panel are those with small probabilities. They contain various class-unrelated objects and can lead to noisy features for constructing class prototypes.

5.5. Performance with Different Semantic Embedding

We use CLIP features in our main experiments. The performance of our method trained with Word2Vec [45] features are shown in Table 3. Note that CLIP model is trained with 400M pairs (image and its text title) collected from the web while Word2Vec is trained with only text data. Our model outperforms state-of-the-art methods in both cases.

6. Limitations and Discussion

We propose a feature generation method using a conditional VAE model. Here we focus on modeling the distribution of the representative samples rather than the whole data distribution. To accomplish that, we propose a sample selection method to collect a set of strictly representative training samples for training our VAE model. We show that our method brings consistent performance improvements over multiple baselines and achieves state-of-the-art performance on both miniImageNet and tieredImageNet datasets. Our method requires a pre-trained NLP model to obtain the semantic embedding of each class. It might also inherit some potential biases from the textual domain. Note that our method does not aim to generate diverse data with large intra-class variance [35, 78]. Building a system that can generate both representative and non-representative samples can greatly benefit various downstream computer vision tasks and is an interesting direction to extend our work.

Acknowledgements. Jingyi Xu is partially supported by a research grant from Zebra Technologies and the SUNY2020 ITSC grant. Hieu Le is funded by Amazon Robotics to attend the conference. We thank Tran Truong, Kien Huynh, and Bento Gonçalves for proofreading the paper.
References


[73] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018. 2


[77] Jingyi Xu, Hieu Le, Mingzhen Huang, ShahRukh Athar, and Dimitris Samaras. Variational feature disentangling for fine-grained few-shot classification. In ICCV, 2021. 2, 8

