Generating Diverse and Reliable Features for Few-Shot Learning

A Dissertation Presented

by

Jingyi Xu

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

Stony Brook University

August 2024

Abstract

Machine learning has achieved remarkable success in data-intensive applications, yet it still encounters challenges when data is insufficient. To tackle this issue, few-shot learning (FSL) has been proposed. FSL aims to develop methods that can rapidly generalize to new tasks with minimal labeled samples. It is particularly beneficial in scenarios where acquiring labeled data is difficult or impractical.

In this thesis, we explore the use of synthetic data generation techniques to enhance FSL. To achieve this, we employ generative models to capture the feature distribution within a continuous latent space, which allows the sampling of new features to facilitate learning in data-scarce scenarios. We demonstrate the capability of this framework to synthesize diverse and reliable features that can enhance FSL across various tasks, including few-shot image classification, fine-grained few-shot classification, few-shot object detection, and class-agnostic object counting. Specifically, for few-shot image classification, we propose a method to generate reliable features via sample selection. For fine-grained few-shot classification, we propose sampling diverse features representing the distribution of intra-class variance. For few-shot object detection, we focus on how to generate features with increased crop-related diversity. For class-agnostic object counting, we present a method for generating reliable features that can serve as object templates. Finally, to conclude this thesis, we investigate a more general question: under what conditions do generative models produce high-quality samples? To address this, we introduce a method for quality assessment based on latent space analysis, ensuring a more reliable use of generated samples beyond FSL scenarios.

Contents

Chapter 1

Introduction

1.1 Overview

Deep learning has achieved significant success in numerous computer vision tasks with sufficiently large-scale labeled training data. However, in many real-world scenarios, annotated data can be hard and costly to obtain, such as rare medical conditions, rare animal species, satellite images, or failure cases in autonomous driving systems. Recently, there has been increasing attention on the development of few-shot learning (FSL) [119, 131, 159, 181], which aims to design models capable of solving tasks using just a few or even zero labeled examples.

In this thesis, we demonstrate that generative models can effectively enhance FSL through data generation. Generative models capture the underlying data distribution by modeling it in a unified latent space. Sampling from this latent space enables the creation of new samples, which can be utilized as additional training data for FSL. Throughout this thesis, we explore multiple strategies to enhance the diversity and reliability of generated samples to facilitate FSL in various scenarios.

Specifically, we achieve this by manipulating either 1) the data distributions used to train generative models or 2) the structure of the latent space itself. These manipulations guide generative models to generate data that optimally supports various FSL tasks. For instance, training generative models exclusively on "easy-to-classify" samples generates reliable data for constructing class prototypes in few-shot classification (see Chapter 3). In another scenario, reorganizing the latent space facilitates the generation of data with increased difficulty levels, thereby enhancing the performance of few-shot object detection tasks (see Chapter 5). Through these strategies, we demonstrate the versatility and effectiveness of generative modeling in enhancing different FSL tasks.

Finally, to conclude this thesis, we aim to address a more general question: under what conditions do generative models produce high-quality samples? To address this, we propose a method for quality assessment based on latent space analysis, ensuring a more reliable use of generated samples beyond FSL scenarios.

1.2 Contributions

Our first contribution focuses on generating reliable data for few-shot image classification. In this task, feature representations from few-shot classes are often biased and fail to accurately represent the real data distribution. We demonstrate that by training a generative model exclusively on representative, i.e., "easy-to-classify" samples, the model will subsequently generate strictly representative samples. These generated samples effectively augment the original training set and significantly improve classification accuracy. Experimental results show that our method enhances three few-shot learning baseline methods by substantial margins, achieving state-of-the-art few-shot classification performance on *mini*ImageNet and *tiered*ImageNet datasets for both 1-shot and 5-shot settings.

In another case, we focus on generating diverse data for fine-grained few-shot classification. Our assumption is that the intra-class variance features, such as backgrounds and illuminations, are shared across different fine-grained classes. Therefore, we propose using a generative model to capture the distribution of intra-class variations from data-sufficient classes. By sampling from this learned distribution, we generate diverse intra-class variance features and incorporate them into the original training set of the few-shot classes for data augmentation. We show that our method significantly outperforms the state-of-the-art methods on multiple challenging fine-grained few-shot image classification benchmarks.

Our third contribution centers on generating data with increased diversity for few-shot object detection. We observe that in object detection, the object proposals often overlap with objects in various ways, resulting in proposals of varying difficulty levels. Training a robust classifier against this crop-related diversity requires abundant training data, which is not available in few-shot settings. To mitigate this issue, we propose a novel generative model architecture, which is capable of generating data with enhanced crop-related diversity. Specifically, we reorganize the latent space such that latent codes with different norms correspond to object crops with different difficulty levels. This reorganization method allows us to generate data with increased crop-related diversity in difficulty levels by adjusting the latent norm. In our experiments, our generated data consistently improve state-of-the-art few-shot object detection methods on benckmark datasets including PASCAL VOC [37] and MS COCO [90].

Furthermore, we utilize generative models to construct reliable class prototypes based on class labels, which enables a new task - counting objects without any annotated exemplars, i.e., zero-shot object counting. Specifically, we leverage the recent advancements in vision-language models, including CLIP [124] and Stable Diffusion [134], to construct class prototypes based on the class name. These prototypes can accurately localize patches containing objects of interests, thereby eliminating the need for human-annotated counting exemplars. Experimental results on a recent class-agnostic counting dataset, FSC-147 [128] validate the effectiveness of our method.

To conclude this thesis, we investigate the conditions under which generative models produce high-quality samples. Our intuition is that the quality of a generated sample directly relates to the amount of the training samples that closely resemble it, and we can infer this information solely by examining the density of the latent space. To this end, we propose a latent density score function for quantifying sample quality. We show that the proposed score highly correlates with the sample quality for various generative models including Variational Autoencoders (VAEs) [72], Generative Adversarial Networks (GANs) [46] and Latent Diffusion Models (LDMs) [134]. Compared to existing methods, our method offers several key advantages including efficiency, generalizability and applicability.

1.3 Organization

The remaining of this thesis is organized as follows. In Chapter 2, we review related approaches in the above aspects. In Chapter 3, we describe our proposed method for generating representative data through sample selection for few-shot image classification. In Chapter 4, we present how to generate data representing the distribution of intra-class variance to diversify fine-grained few-shot categories. In Chapter 5, we focus on generating data with increased crop-related diversity to facilitate few-shot object detection. In Chapter 6, we present a method for generating data that can serve as reliable object templates for class-agnostic object counting. Finally, Chapter 7 details our proposed latent-based quality assessment method, which measures sample quality by examining the latent space of generative models.

1.4 Publications List

Works presented in this thesis are supported by the following list of publications:

1. **Jingyi Xu**, Hieu Le, Mingzhen Huang, ShahRukh Athar, Dimitris Samaras, *Variational Feature Disentangling for Fine-Grained Few-Shot Classification*, **ICCV, 2021**.

2. **Jingyi Xu**, Hieu Le, *Generating Representative Samples for Few-Shot Classification*, **CVPR, 2022**.

3. **Jingyi Xu**, Hieu Le, Dimitris Samaras, *Generating Features with Crop-Related Diveristy for Few-Shot Object Detection*, **CVPR, 2023**.

4. **Jingyi Xu**, Hieu Le, Vu Nguyen, Viresh Ranjan, Dimitris Samaras, *Zero-Shot Object Counting*, **CVPR, 2023**.

5. **Jingyi Xu**, Hieu Le, Dimitris Samaras, *Assessing Sample Quality via the Latent Space of Generative Models*, **Under Review**.

Chapter 2

Literature Review

In this dissertation, we focus on enhancing the diversity and reliability of generated samples to better support downstream vision tasks. We employ variational autoencoders (VAEs) and diffusion models as our generative frameworks. We demonstrate that the generated data can facilitate various tasks, particularly in data-scarce scenarios such as few-shot image classification, fine-grained few-shot classification, few-shot object detection, and class-agnostic object counting. Furthermore, we investigate the conditions under which generative models perform well, ensuring a more reliable use of generated samples.

In this chapter, we will review the literature related to this research work, including VAEs, few-shot classification, few-shot object detection, zero-shot learning, class-agnostic object counting and related quality assessment methods.

2.1 Few-shot Classification

Few-shot classification has been receiving increasing attention over the past few years. Concretely, few-shot classification learns classifiers given only a few labeled examples of each class. In a typical N-way K-shot few-shot setting, we are given a training set D_{train} and a testing set D_{test}. The training set D_{train} contains $I = K * N$ examples from N classes each with K examples. K is often small (*i.e.*, $K = 1$ or $K = 5$). The goal is to learn a classifier on D_{train} which performs well on D_{test}.

The problem of learning from very few examples firstly attracted the attention of E. G. Miller *et al.* in 2000 [107], who postulated a shared density on digit transforms and proposed a Congealing algorithm to bring test digit image into correspondence with class-specific congealed digit image. Thereafter, more and more efforts were devoted to FSL research. The development process of FSL research can be roughly divided into two periods, non-deep period (from 2000 to 2015) and deep period (from 2015 to now). In particular, most of the famous early FSL approaches in non-deep period are based on the generative model. They seek to estimate the joint distribution or the conditional distribution given a supervision, albeit on very few observed training samples and then make predictions for test samples. With deep learning booming, especially the great success achieved by CNNs on visual tasks, many FSL researchers began to shift their sights from non-deep models to deep models. Representative FSL approaches in this deep period can be roughly divided into four main categories: metric-based approaches, optimization-based approaches, augmentation-based approaches and semantics-based approaches.

Metric-based approaches

Metric-based approaches [116, 150, 154, 155, 159, 187, 193] tackle the few-shot learning problem by learning to transform the data to a lower dimension representation and then clustered and compared using a specific distance/metric function.

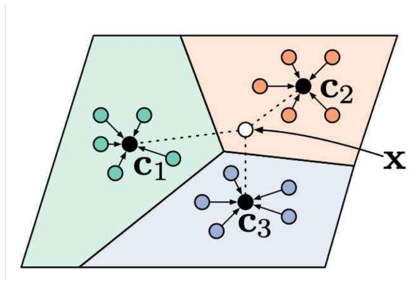


Figure 2.1: Few-shot prototypes c_k are computed as the mean of embedded support examples for each class. Image Source: [150].

ProtoNet [150] is one of the most popular and effective approaches in the FSL literature. It is based on episode training, where an episode consists of randomly selected tasks from the training set with k number of labeled samples from each class. ProtoNet learns to classify samples by comparing the distance to the representatives of each class. The network calculates a prototype representation (an M-dimensional representation) of each class using an embedding function $f_\theta : \mathbb{R}^D \to \mathbb{R}^M$, where θ is the learnable parameters. The prototype is given by $c_k \in \mathbb{R}^M$ which is calculated using the mean vector of the embedded support points in the class space, as shown in Figure 2.1. c_k is given by:

$$c_k = \frac{1}{s_k} \sum_{(x_n, y_n) \in s_k} f_\theta(x_n), \tag{2.1}$$

where s_k is the number of samples in class k. Then Euclidean distance is leveraged to predict the probability:

$$p(y_j = k | x_j) = \frac{\exp(-d(f_\theta(x_j), c_k))}{\sum_{k'=1}^{K} \exp(-d(f_\theta(x_j), c'_k))}, \tag{2.2}$$

where $d(x_i, x_j)$ denotes the Euclidean distance between x_i and x_j and K is the total number of classes.

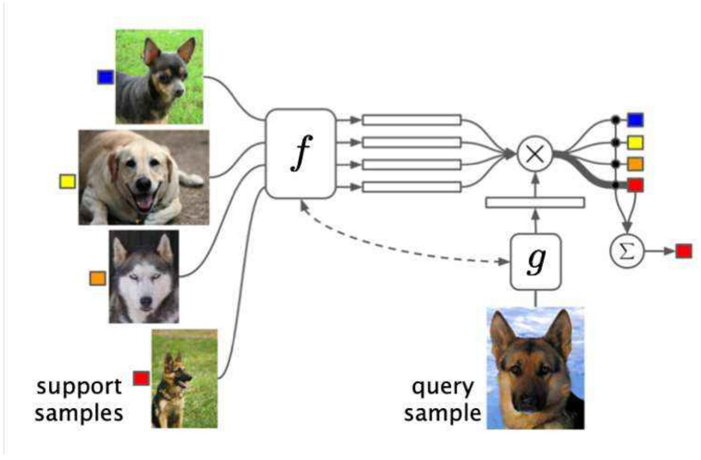


Figure 2.2: **MatchingNet architecture**. Image Source: [159].

Vinyals *et al.* propose **MatchingNet** [159] to accumulate information on a given task with memory mechanism and utilize cosine distance in an attention kernel as measurement. Figure 2.2 shows the overall architecture of MatchingNet. Given a test sample $\hat{x}$, MatchingNet predicts output class $\hat{y}$ as follows:

$$\hat{y} = \sum_{i=1}^{k} a(\hat{x}, x_i) y_i, \tag{2.3}$$

where x_i, y_i are the samples and labels from the support set $\mathrm{S} = \{(x_i, y_i)\}_{i=1}^{k}$, and a is an attention mechanism which is the softmax over the cosine distance c:

$$a(\hat{x}, x_i) = \frac{e^{c(f(\hat{x}), g(x_i))}}{\sum_{j=1}^{k} e^{c(f(\hat{x}), g(x_j))}}, \tag{2.4}$$

where f and g are embedding functions to embed $\hat{x}$ and x_i (potentially with $f = g$).

Unlike ProtoNet and MatchingNet which use the non-parametric Euclidean distance or cosine distance to measure the similarity between pairwise features, **RelationNet** [154] adopts a learnable CNN to measure pairwise similarity. The CNN takes the concatenation of the feature maps of support sample x_i and query sample x_j as input and outputs the relation score $r(x_i, x_j)$, as shown in Figure 2.3.

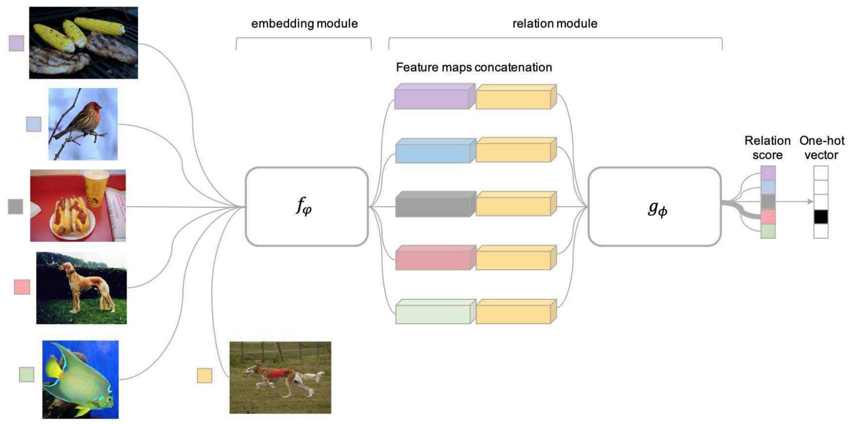


Figure 2.3: **RelationNet architecture**. Image Source: [154].

Optimization-based approaches

Optimization-based [42, 83, 88, 89, 125, 139, 143] suggest to fine-tune a base learner for task T using its few support samples and make the base learner converge fast on these samples within several parameter update steps. Generally, optimization-based approaches contain a base-learner and a meta-learner. Let θ_b and θ_m denote the parameters of base-learner and meta-learner, respectively. The learning process occurs at two levels, *i.e.*, gradual learning and rapid learning. Gradual learning is performed across tasks, which aims to optimize the meta-learning parameters θ_m. The rapid learning of the base-learner for each specific task θ_b can be facilitated by the gradual learning.

MAML is a meta-learning framework which has a profound influence on the field of FSL. Its core idea is to search for a good parameter initialization for θ_b by cross-task training strategy such that the base-learner with this initialization can rapidly generalize new tasks using a few support samples. Algorithm 1 depicts the overall learning process of the MAML approach.

Algorithm 1 MAML process summarization.

```
Require: Task distribution $p(\tau)$, step size $\gamma$, learning rate $\beta$
  Initialize random values for $\theta$
  while not done do
    Perform task batch $\tau_i \sim p(\tau)$
    for all $\tau_i$ do
      Evaluate $\nabla\theta L_{\tau_i}(f_\theta)$ wrt. $k$ samples
      Calculate the adapted parameters:
      $\theta_i' = \theta - \gamma * \nabla\theta L_{\tau_i}(f_\theta)$
    end for
    Update $\theta \leftarrow \theta - \beta * \nabla\theta \sum_{\tau_i \sim p(\tau)} L_{\tau_i}(f_\theta)$
  end while
```

Another representative optimization-based approach is **Meta-Learner LSTM** [129]. It is based on a Long Short-Term Memory meta-learner (LSTM) to fine-tune the base-learner on the few support samples. As shown in Figure 2.4, the LSTM-based meta-learner takes as input the loss and gradient of the base-learner with respect to each support sample. Its hidden state is treated as the updated base-learner parameter. In this framework, the vanilla gradient-based optimization for base-learner parameters is superseded by an LSTM in the hope of learning appropriate parameter updates specifically for the scenario where a few updates will be made.

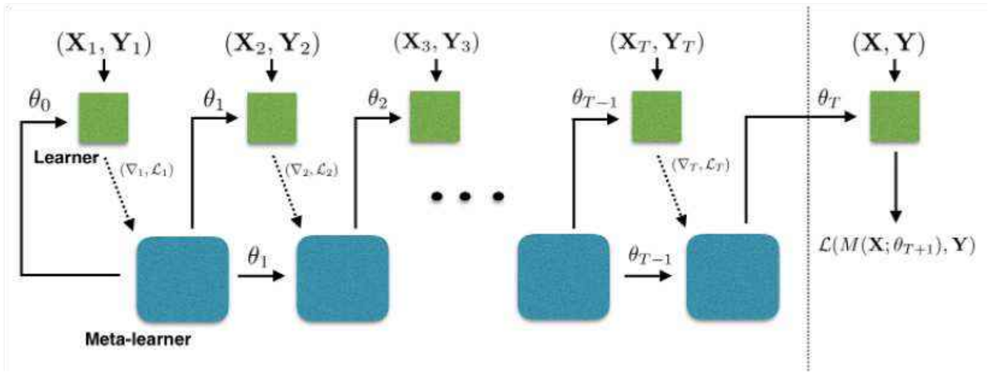

$(\mathbf{X}_1, \mathbf{Y}_1)$
$(\mathbf{X}_2, \mathbf{Y}_2)$
$(\mathbf{X}_3, \mathbf{Y}_3)$
$(\mathbf{X}_T, \mathbf{Y}_T)$
$(\mathbf{X}, \mathbf{Y})$
θ_0
θ_1
θ_2
θ_{T-1}
θ_T
Learner
$(\nabla_1, \mathcal{L}_1)$
$(\nabla_2, \mathcal{L}_2)$
$(\nabla_T, \mathcal{L}_T)$
θ_1
θ_{T-1}
$\mathcal{L}(M(\mathbf{X}; \theta_{T+1}), \mathbf{Y})$
Meta-learner

Figure 2.4: Computational graph for the forward pass of the meta-learner. The dashed line divides examples from the training set and test set. Each (X_i, X_i) is the i th batch from the training set whereas (X, Y) is all the elements from the test set. The dashed arrows indicate that the back-propagation is disabled when training the meta-learner. M denotes the learner and $M(X;\theta)$ is the output of learner M using parameters θ for inputs X. Image Source: [129].

Augmentation-based approaches

Augmentation-based approaches aim to compensate for the insufficient number of available samples by generation. Most methods use the idea of Generative Adversarial Networks (GANs) or autoencoder to generate samples or features to augment the training set.

Δ−**Encoder** [141] learns to extract transferable intra-class variation (called Δ) from the base classes and apply this variation to the novel classes so as to synthesize additional samples for the novel classes. It develops an encoder-decoder network whose bottleneck embedding is expected to capture the intra-class variation Δ.

Another representative augmentation-based FSL method is **Hallucinator** [165] proposed by Wang *et al.* As shown in Figure 2.5, Hallucinator

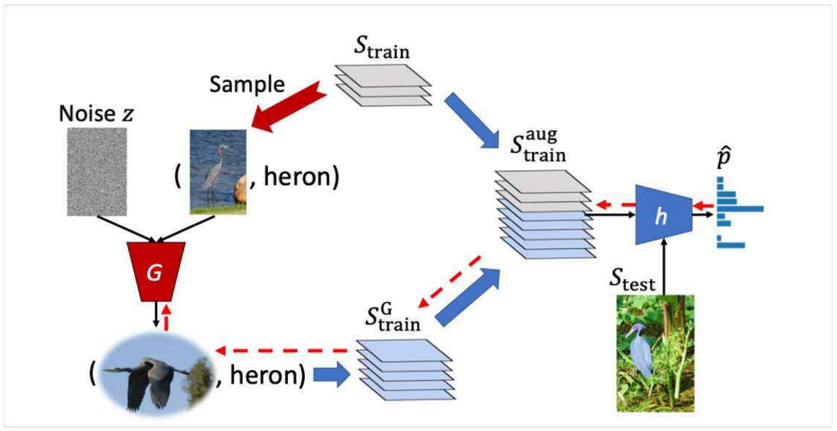


Figure 2.5: Meta-learning with hallucination. The initial training set S_{train} is augmented by a set of generated samples S_{train}^{G}. S_{train}^{G} is obtained by sampling real seed examples and noise vectors z and passing them to a parametric hallucinator G. The hallucinator is trained end-to-end along with the classification algorithm h. Dotted red arrows indicate the flow of gradients during back-propagation. Image Source: [165].

uses an MLP-based generator G to augment features for the support set, *i.e.*, $\hat{f} = G(f, z)$, where f is an original feature and z is a noise vector. The proposed generator can be incorporated into a variety of meta-learners and provides significant gains.

Dual TriNet [29] leverages semantic space to learn the transformation between the image features at multiple layers and the semantic space. In semantic space, they search for related concepts, which are then projected back into the image feature spaces by the decoder. They explore two strategies to augment the semantic space and these strategies result in complex augmented feature distributions in the image feature space, leading to substantially better performance.

Semantic-based approaches

For semantic-based methods, the semantics of the data are used along with the prior knowledge for the model to either learn or optimize to novel categories.

Xing *et al.* proposed **AM3** [179], which adaptively combine informa-

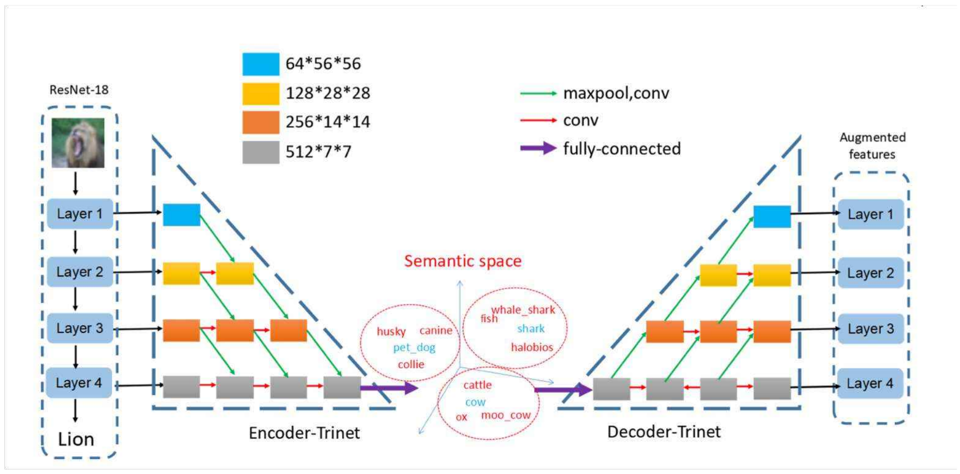


Figure 2.6: Overview of the Dual TriNet architecture. The image features are extracted by ResNet-18 and augmented features are generated by dual TriNet. Encoder TriNet projects features to the semantic space. After augmenting data in semantic space, the decoder TriNet is used to obtain the corresponding augmented features. Both real and augmented data are used to train the classification model. Image Source: [29].

tion from both visual and semantics modalities according to new image categories to be learned. Specifically, the prototype representation is modeled as a convex combination of the visual and the semantic feature representations:

$$\mathrm{p}'_c = \lambda_c * \mathrm{p}_c + (1 - \lambda_c) * w_c,$$

where λ_c is the adaptive mixture coefficient (conditioned on the category) and $w_c = g(e_c)$ is a transformed version of the label embedding for class c. The representation e_c is the pre-trained word embedding of label c. The coefficient λ_c is conditioned on category and calculated as follows:

$$\lambda_c = \frac{1}{1 + \exp(-h(w_c))},$$

where h is the adaptive mixing network, with parameters θ_h. Figure 2.7 illustrates the proposed model.

Another representative semantics-based approach is **KTN** [119], which jointly incorporates visual feature learning, knowledge inferring and clas-

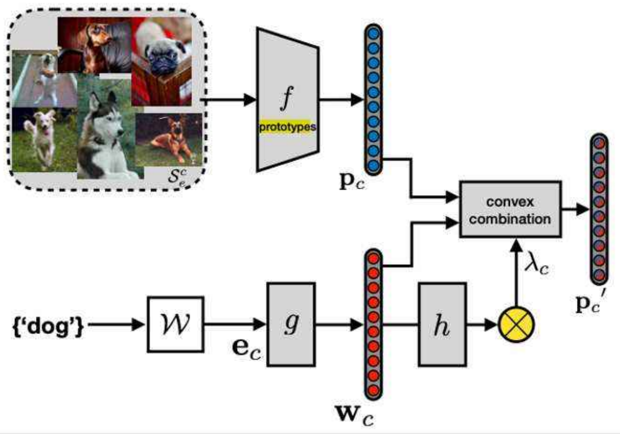


Figure 2.7: Adaptive modality mixture model. The final category prototype is a convex combination of the visual and the semantic feature representations. The mixing coefficient is conditioned on the semantic label embedding. Image Source: [179].

sifier learning into one unified framework, as shown in Figure 2.8. Specifically, a visual feature extractor based on Convolutional Neural Network (CNN) is trained by optimizing cosine similarity with the training data of base categories, which is used to extract the representation of examples and generate vision-based classifiers of novel categories. To well leverage the prior knowledge, a semantic-visual mapping network (M-Net) is developed to conduct knowledge inference and the semantic relationship of all categories is explicitly explored by employing the graph convolutional network and knowledge graph. This mapping can serve as the knowledge-based classifiers generator of novel categories. Finally, an adaptive fusion scheme is proposed to infer the final classifiers by integrating the above two classifiers.

2.2 Few-Shot Object Detection

Few-shot object detection aims at detecting novel objects with only few annotated instances. A number of prior methods [39–41, 50, 51, 66,

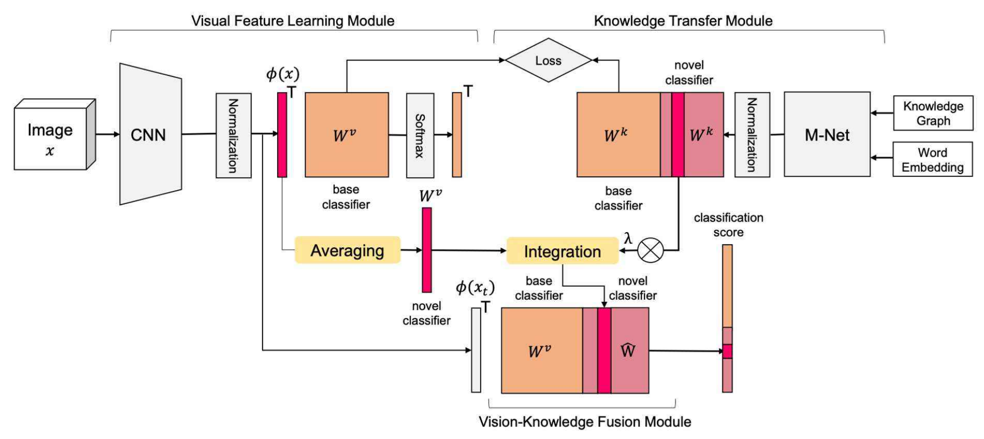


Figure 2.8: Illustration of the proposed Knowledge Transfer Network architecture (KTN) for few-shot image recognition. Image Source: [119].

66, 78, 87, 99, 100, 121, 152, 169–171, 200] have been proposed to address this challenging task. One line of work focuses on the **meta-learning** paradigm, which has been widely explored in few-shot classification [38, 62, 142, 166, 177, 182, 185, 186]. Meta-learning based approaches introduce a meta-learner to acquire meta-knowledge that can be then transferred to novel classes. [62] propose a meta feature learner and a reweighting module to fully exploit generalizable features from base classes and quickly adapt the prediction network to predict novel classes. [166] propose specialized meta-strategies to disentangle the learning of category-agnostic and category-specific components in a CNN based detection model. Meta R-CNN [182] extends Faster / Mask R-CNN by proposing meta-learning over RoI (Region-of-Interest) features.

Another line of work adopts a **two-stage fine-tuning** strategy and has shown great potential recently [18, 122, 152, 162, 172]. [162] propose to fine-tune only box classifier and box regressor with novel data while freezing the other paramters of the model. This simple stragetegy outperforms previous meta-learners. FSCE [152] leverages a contrastive proposal encoding loss to promote instance level intra-class compactness and inter-class variance. FADI [18] associates each novel category to one base category and then the network is trained to align the feature distribution of the novel category to the associated base category. Guirguis *et al* [47] propose a constraint-based finetuning approach (CFA) to alleviate catastrophic forgetting, while achieving competitive results without increasing the model capacity. Fan *et al* [41] propose a few-shot detector without forgetting, Retentive R-CNN, which can assist novel class adaptation with base class knowledge and ensemble base and novel class detectors.

Other **data augmentation** works try to increase the variance of the data for novel categories. Zhang *et al* [197] introduce a hallucinator network that learns to generate additional training examples for novel categories. The features in the RoI head of novel category samples are augmented by leveraging the shared within-class feature variation from base categories. Kaul *et al* [66] show in their experiments that data augmentation, *i.e.*, color jittering, random cropping, mosaicing, and dropout for the extracted features for each RoI, significantly improves the performance.

2.3 Variational Autoencoder

Variational Autoencoder (VAE) can be regarded as a mixture of an encoder and a decoder Bayesian network. The encoder maps an input data (*e.g.*, an image) x to a latent vector z, and then, the decoder maps the latent vector z back to image or data space. The true posterior distribution $p(z|x)$ is approximated with another distribution $q(z_V|X)$. The Kullback-Leibler divergence between the true distribution and the approximation is:

$$KL[q(z|x)||p(z|x)] = \int_Z q(z|x)\log\frac{q(z|x)}{p(z|x)}. \tag{2.5}$$

Since the Kullback-Leibler divergence is always greater than or equal to zero, maximizing the marginal likelihood $p(x)$ is equivalent to maximizing the evidence lower bound (ELBO) defined as follows:

$$ELBO = E_{q(z|x)}[\log p(x|z)] - KL(q(z|x)||p(z)). \tag{2.6}$$

Different VAE variants have been proposed to generate diverse data [49, 58, 73, 144]. β-VAE [58] imposes a heavy penalty on the KL divergence term to enhance the disentanglement of the latent dimensions. By traversing the values of latent variables, β-VAE can generate data with disentangled variations. ControlVAE [144] improves upon β-VAE by introducing a controller to automatically tune the hyperparameter added in the VAE objective. However, disentangled representation learning can not capture the desired properties without supervision. Some VAE methods allow explicitly controllable feature generation including CSVAE [73] and PCVAE [49]. CSVAE [73] learns latent dimensions associated with binary properties. The learned latent subspace can easily be inspected and independently manipulated. PCVAE [49] uses a Bayesian model to inductively bias the latent representation. Thus, moving along the learned latent dimensions can control specific properties of the generated data.

Using a conditional VAE to model a feature distribution has been used before in many computer vision tasks such as image classification [69, 140, 181, 194], image generation [36, 95], image restoration [34], or video processing [118]. Using VAE models for generating features conditioned on the corresponding semantic embedding is fairly common in zero-shot learning (ZSL) methods[9, 48, 109, 140, 191, 195]. Mishra *et al* [109] 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* [5] show that a conditional VAE can be used together with a GAN system to synthesize images for unseen classes effectively. Keshari *et al* [67] focus on generating a specific set of *hard* samples which are closer to another class and the decision boundary.

2.4 Zero-shot Learning

Zero-shot learning (ZSL) is also closely related to FSL, which aims to address the novel class categorizations without any labeled samples. The key idea is to learn a mapping function between the semantic and the visual space on the base classes, then apply the mapping to categorize novel classes. The semantic spaces in ZSL are typically attribute-based [160], text description-based [130], and word vector-based [43].

Using VAE models for generating features conditioned on the corresponding semantic embedding is fairly common in ZSL methods[9, 48, 109, 140, 191, 195]. Mishra [109] propose to use a conditional VAE to learn the underlying probability distribution of the image features conditioned on the class embedding vector. Specifically, the input x and the semantics class embedding vector A_y are concatenated and passed through an encoder. The encoder generates the probability distribution $q(z|x, A_y)$

which is assumed to be an isotropic Gaussian. The sampled z is projected to the image space to reconstruct the original x by a decoder. Once the CVAE is properly trained, one can use the decoder part of the model to generate any number of samples of a particular class using a simple algorithm: Sample z from a standard normal, concatenate A_y, and pass it through the decoder.

Arora [5] also synthesize exemplars from the unseen classes using a CVAE-based architecture for ZSL. Moreover, the architecture is further coupled with a discriminator (a multivariate regressor) that learns a mapping from the VAE generator's output to the class attribute, as shown in Figure 2.9. This feedback helps to improve the generator by encouraging it to generate exemplars that are of highly discriminative nature. They show that such an explicit feedback driven mechanism yields much better prediction accuracies compared to a vanilla conditional generative model.

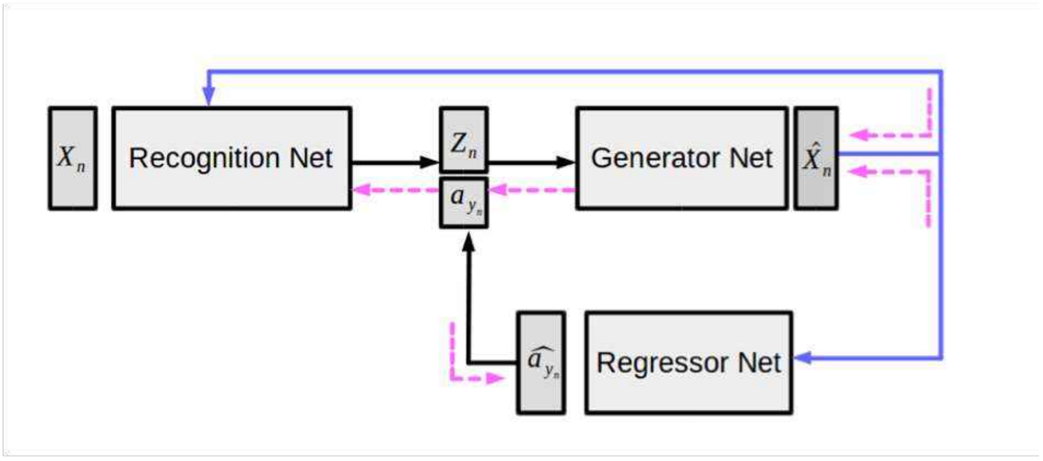


Figure 2.9: The architecture for zero-shot set-up proposed by [5]. Each block represents a feed-forward neural network. The encoder to z_n link is stochastic similar to a VAE. The blue lines direct feedback connection into regressor and recognition network for the generated X^n. The red-lines represent the back propagation direction. Image Source: [5].

Keshari *et al.* [67] further improves CVAE-based FSL performance by introduing the concept of Over-Complete Distribution (OCD). The objective of over-complete distributions is to generate challenging samples that are closer to other classes, which consequently helps in increasing the generalizability of the network with respect to the unseen classes. In addition, they propose to incorporate Online Batch Triplet Loss (OBTL) to enforce separability between classes and Center Loss (CL) to reduce the spread within the class. The proposed framework is shown in Figure 2.10. It uses CVAE with encoder and decoder modules. The output of CVAE is given to the regressor where regressor maps the generated samples to its respective attributes. To generate the unseen synthetic data, attributes of unseen samples and randomly sampled z are provided to the trained decoder.

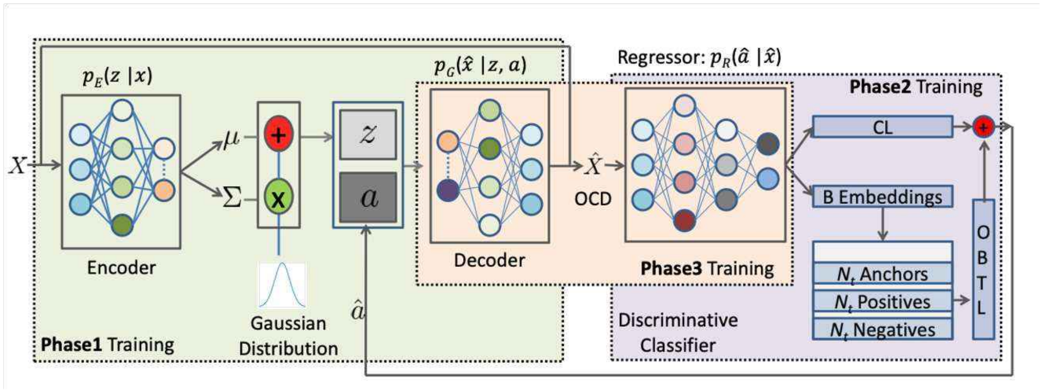


Figure 2.10: Illustration of the proposed OCD-CVAE framework. The framework uses CVAE with encoder $p_E(z|x)$ and decoder $p_G(\hat{x}|z,a)$ modules. The output of CVAE is given to the regressor $p_R(\hat{a}|\hat{x})$ where regressor maps the generated samples to its respective attributes. To generate the unseen synthetic data, attributes of unseen samples and randomly sampled z are provided to the trained decoder. Image Source: [67].

2.5 Class-agnostic Object Counting

Class-agnostic object counting aims to count arbitrary categories given only a few exemplars [6, 45, 93, 97, 113, 128, 146, 184, 189]. GMN [97] uses a shared embedding module to extract feature maps for both query images and exemplars, which are then concatenated and fed into a matching module to regress the object count. FamNet [128] adopts a similar way to do correlation matching and further applies test-time adaptation. These methods require human-annotated exemplars as inputs. Recently, Ranjan *et al.* have proposed RepRPN [127], which achieves exemplar-free counting by identifying exemplars from the most frequent objects via a Region Proposal Network (RPN)-based model. However, the class of interest can not be explicitly specified for the RepRPN. In comparison, our proposed method for class-agnostic object counting 6 can count instances of a specific class given only the class name.

2.6 Quality Assessment Metrics

Previous metrics for quality assessment can be grouped into two main categories: model-wise evaluation metrics and instance-wise evaluation metrics. Model-wise evaluation metrics measure the performance of different generative models, while instance-wise evaluation metrics aim to compare the quality of each individual generated sample.

Prevalent model-wise metrics include Inception Score (IS) [137], Kernel Inception Distance (KID) [12] and Frechèt Inception Distance (FID) [57]. They quantify the performance of a generative model by measuring the distribution discrepancy between the generated samples and real samples in a high-dimensional feature space. Sajjadi *et al.* [136] propose to further disentangle this discrepancy between distributions into two components: precision and recall. Precision represents the quality of generated samples while recall corresponds to the coverage of the real target distribution. Naeem *et al.* [112] improve upon precision and recall by introducing density and coverage: density improves upon precision by being more robust to outliers and coverage improves upon recall by preventing the overestimation of the latent manifold. Although the above metrics have demonstrated their effectiveness in assessing generative models, they are not suitable to measure individual sample quality since they work on a set of generations.

Unlike model-wise metrics, instance-wise metrics are applied on individual generated samples for performance evaluation. They are helpful for users to select samples from generative models, which might produce noisy, unrealistic samples with artifacts, especially for underrepresented cases [82] such as rare categories or extreme object poses. The realism score [74] measures the perceptual quality of individual samples by estimating how close a given fake sample is to the latent manifold of real samples. Recently, Han *et al.* have proposed the rarity score [52], which measures how rare a synthesized sample is based on the real data distribution. Our proposed method and rarity score share the spirit of estimating the density around the target fake sample on the real manifold. Nevertheless, rarity score defines this manifold using a pre-trained classification network, *e.g.*, VGG16, while our method directly leverages the latent manifold of the generative models themselves. We show that in this latent manifold, the density correlates with the perceptual quality of the generated samples.

Chapter 3

Generating Representative Data for Few-Shot Classification

3.1 Overview

In this Chapter, we focus on generating reliable data for few-shot image classification. Few-shot image classification 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 [117, 161], rare animal species [167], failure cases in autonomous systems [102, 103, 133]). 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 [164].

Many FSL methods [24, 69, 96, 150, 150, 179, 192] 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 [94, 183]. An effective strategy to enhance the representativeness of the prototype is to employ textual semantic embeddings learned via NLP models[31, 108, 120, 123] using large unsupervised text corpora [179, 192]. These semantic embeddings implicitly associate a class name, such as "Yorkshire Terriers", with the class repre-

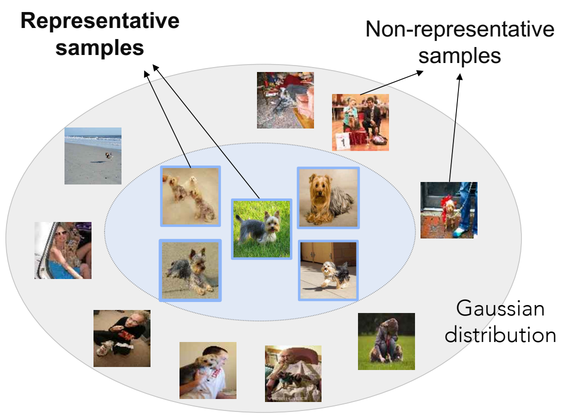


Figure 3.1: **Representative Samples.** We refer representative samples to the "easy-to-recognize" samples that faithfully reflect the key characteristics of the category. We identify those samples and then use them to train a VAE model for feature generation, conditioned on class-representative semantic embeddings. We show that the generated data significantly improves few-shot classification performance.

sentative semantic attributes such as "smallest dog" or "long coat" [1] (Fig. 6.1), 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* [192] 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* [179] propose a method that computes an adaptive mixture coefficient to combine features from the visual and textual 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 work, we propose a novel method to obtain class-representative prototypes. Inspired by zero-shot learning (ZSL) methods[9, 48, 195], we propose to generate visual features via a variational autoencoder (VAE) model [151] 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 [5, 109]. 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 *mini*ImageNet and *tiered*ImageNet datasets. It works exceptionally well for 1-shot scenarios where our method outperforms state-of-the-art methods[168, 187] by $5 \sim 6\%$ 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. 3.1). The embeddings of these representative samples often lie close to their corresponding class centers, which are particularly useful for constructing class-representative prototypes.

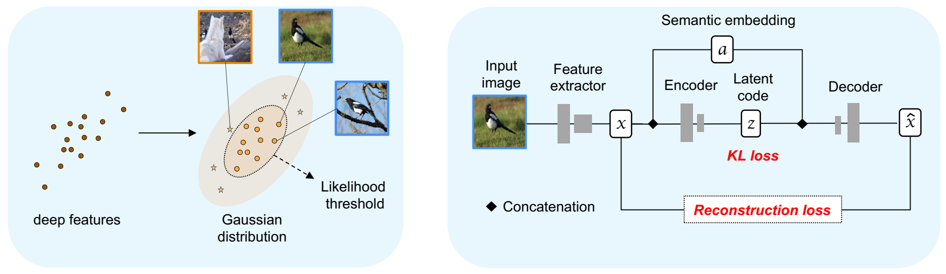
Specifically, we guide the VAE model to generate representative samples by selecting only representative data from the base classes for training it. In essence, our VAE model is trained to model the data distribution of the training set. As the training set contains only representative data, the trained 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 *mini*Imagenet dataset, to obtain the best results. Our analyses show that this approach consistently improves the FSL classification performance by $1 \sim 2\%$ across all benchmarks for three different baselines[24, 96, 150].

The main contributions in this Chapter 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, *tiered*ImageNet and *mini*-ImageNet.



(a) Sample Selection method **(b) Conditional VAE model**

Figure 3.2: **Overview –** The key aspect of our approach is to subset our training set to the most representative samples to train a conditional VAE model that generates more representative features. **(a)** To select representative samples, we assume that the features of each class follow a multivariate Gaussian distribution. We estimate the distribution parameters and compute a probability for each data point belonging to the class distribution. We identify a set of representative samples by setting a threshold on the probability. **(b)** We train a VAE to generate visual features, conditioned on the semantic embedding of each class. Using only representative samples (the output of the sample selection step) to train this VAE model improves the representativeness of the generated samples.

3.2 Proposed Method

3.2.1 Problem Definition

In a typical few-shot classification setting, we are given a set of data-label pairs $D = \{(x^i, y^i)\}$. Here $x^i \in R^d$ 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.

3.2.2 Overall Pipeline

Fig. 3.2 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 [24], ProtoNet [150] and E3BM [96] by large margins.

Class-representative Sample Selection

In this work, we are interested in representative samples as they can serve as reliable data points for constructing a class-representative prototype[24, 150]. 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:

$$\mu^i = \frac{1}{n^i} \sum_{j=1}^{n^i} x^j, \tag{3.1}$$

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 Σ^i for the distribution of class i is calculated as:

$$\Sigma^i = \frac{1}{n^i - 1} \sum_{j=1}^{n^i} (x^j - \mu^i)(x^j - \mu^i)^T. \tag{3.2}$$

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:

$$p(x^j|\mu^i, \Sigma^i) = \frac{\exp\{-\frac{1}{2}(x^j - \mu^i)^T {\Sigma^i}^{-1} (x^j - \mu^i)\}}{(2\pi)^{k/2}|\Sigma^i|^{1/2}}, \tag{3.3}$$

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 ϵ on the estimated probability, we filter out those samples with small probabilities and get a set of representative features for class i:

$$\mathbb{D}^i = \{x^j \mid p(x^j|\mu^i, \Sigma^i) > \epsilon\}, \tag{3.4}$$

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

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[151] (see Fig. 3.2b). 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:

$$\begin{aligned} L_V(x^j) = & \mathrm{KL}\left(q(z|x^j,a^i) \| p(z|a^i)\right) \\ & - \log p(x^j|z,a^i), \end{aligned} \tag{3.5}$$

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:

$$L_V = \sum_{i=1}^{C_b} \sum_{x \in \mathbb{D}^i} L_V(x) \tag{3.6}$$

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:

$$\mathbb{G}^y = \{\hat{x} | \hat{x} = G(z, a^y), z \sim \mathcal{N}(0,I)\}. \tag{3.7}$$

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:

$$\mathrm{p}^y = w_g * \frac{1}{|\mathbb{G}^y|} \sum\nolimits_{\hat{x}^j \in \mathbb{G}^y} \hat{x}^j + w_s * \frac{1}{|\mathbb{S}^y|} \sum\nolimits_{x^j \in \mathbb{S}^y} x^j, \tag{3.8}$$

Method	Backbone	*mini*ImageNet		*tiered*ImageNet	
		1-shot	5-shot	1-shot	5-shot
Matching Net [159]	ResNet-12	65.64 ± 0.20	78.72 ± 0.15	68.50 ± 0.92	80.60 ± 0.71
MAML [42]	ResNet-18	64.06 ± 0.18	80.58 ± 0.12	-	-
SimpleShot [163]	ResNet-18	62.85 ± 0.20	80.02 ± 0.14	69.09 ± 0.22	84.58 ± 0.16
CAN [60]	ResNet-12	63.85 ± 0.48	79.44 ± 0.34	69.89 ± 0.51	84.23 ± 0.37
S2M2 [104]	ResNet-18	64.06 ± 0.18	80.58 ± 0.12	-	-
TADAM [116]	ResNet-12	58.50 ± 0.30	76.70 ± 0.30	62.13 ± 0.31	81.92 ± 0.30
AM3 [179]	ResNet-12	65.30 ± 0.49	78.10 ± 0.36	69.08 ± 0.47	82.58 ± 0.31
DSN [148]	ResNet-12	62.64 ± 0.66	78.83 ± 0.45	66.22 ± 0.75	82.79 ± 0.48
Variational FSL [194]	ResNet-12	61.23 ± 0.26	77.69 ± 0.17	-	-
MetaOptNet [83]	ResNet-12	62.64 ± 0.61	78.63 ± 0.46	65.99 ± 0.72	81.56 ± 0.53
Robust20-distill [35]	ResNet-18	63.06 ± 0.61	80.63 ± 0.42	65.43 ± 0.21	70.44 ± 0.32
FEAT [187]	ResNet-12	66.78 ± 0.20	82.05 ± 0.14	70.80 ± 0.23	84.79 ± 0.16
RFS [155]	ResNet-12	62.02 ± 0.63	79.64 ± 0.44	69.74 ± 0.72	84.41 ± 0.55
Neg-Cosine [92]	ResNet-12	63.85 ± 0.81	81.57 ± 0.56	-	-
FRN [168]	ResNet-12	66.45 ± 0.19	82.83 ± 0.13	71.16 ± 0.22	86.01 ± 0.15
Meta-Baseline [24]	ResNet-12	63.17 ± 0.23	79.26 ± 0.17	68.62 ± 0.27	83.29 ± 0.18
Meta-Baseline + SVAE (Ours)	ResNet-12	69.96 ± 0.21	79.92 ± 0.16	73.05 ± 0.24	83.96 ± 0.18
Meta-Baseline + R-SVAE (Ours)	ResNet-12	72.79 ± 0.19	80.70 ± 0.16	73.90 ± 0.24	84.17 ± 0.18
ProtoNet [187]	ResNet-12	62.39	80.53	68.23	84.03
ProtoNet + SVAE (Ours)	ResNet-12	73.01 ± 0.24	83.13 ± 0.40	76.36 ± 0.65	85.65 ± 0.50
ProtoNet + R-SVAE(Ours)	ResNet-12	**74.84 ± 0.23**	**83.28 ± 0.40**	76.98 ± 0.65	85.77 ± 0.50
E3BM [96]	ResNet-12	64.09 ± 0.37	80.29 ± 0.25	71.34 ± 0.41	85.82 ± 0.29
E3BM + SVAE (Ours)	ResNet-12	73.07 ± 0.39	80.82 ± 0.31	79.85 ± 0.43	86.82 ± 0.32
E3BM + R-SVAE(Ours)	ResNet-12	73.35 ± 0.37	80.95 ± 0.31	**80.46 ± 0.43**	**86.99 ± 0.32**

Table 3.1: **Comparison to prior works on *mini*ImageNet and *tiered*ImageNet**. Average 5-way 1-shot and 5-way 5-shot accuracy (%) with 95% confidence intervals. SVAE denotes our method using the VAE trained with all features in the base set. R-SVAE denotes the one trained with only representative features. The **best** performance is highlighted in bold.

where $\mathbb{S}^y$ 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 3.4.2). Compared to the methods that correct the original biased prototype, our model does not require any carefully designed combination scheme.

3.3 Experiments

3.3.1 Experimental Settings

Datasets. We evaluate our method on two widely-used benchmarks for few-shot learning, *mini*ImageNet [159] and *tiered*ImageNet [131]. ***mini*ImageNet** is a subset of the ILSVRC-12 dataset [30] It contains 100 classes and each class consists of 600 images. The size of each image is 84×84. Following the evaluation protocol of [129], we split the 100 classes into 64 base classes, 16 validation classes, and 20 novel classes for pre-training, validation, and testing. ***tiered*ImageNet** 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 [187], Meta-Baseline [24] and E3BM [96]. ProtoNet is known as a strong and classic prototypical approach. In our experiments, we use the ProtoNet implementation of Ye *et al* [187]. Meta-Baseline [24] uses a ProtoNet model to fine-tune a generic classifier via meta-learning. E3BM [96] 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.

3.3.2 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 [187], 512 for Meta-Baseline [24], and 640 for E3BM[96]. 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 [55] are the nonlinear 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 [123]. We empirically set the combination weights $[w_g, w_s]$ in Equation 3.8 to $[\frac{1}{2}, \frac{1}{2}]$ 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 3.4.1.

3.3.3 Results

Table 7.1 presents the 5-way 1-shot and 5-way 5-shot classification results of our methods on *mini*ImageNet and *tiered*ImageNet in comparision with previous FSL methods. Here all methods use ResNet12/ResNet18 architectures as feature extractors with input images of size 84×84. Thus, the comparison is fair. For the rest of this chapter, we denote our VAE trained with all data as **SVAE** (Semantic-VAE) and the model trained with only representative data as **R-SVAE** (Representative-SVAE).

We apply our methods on top of the Meta-Baseline [24], ProtoNet[187], and E3BM[96]. Our methods consistently improve all three baselines under all settings and for all datasets. They work particularly well under the 1-shot settings, in which sample bias is a more pronounced issue. Using the model trained on all data - SVAE, we report 6.8% $\sim$ 10% 1-shot accuracy improvements for all three baselines. Our 1-shot performance for all the baselines outperforms the state-of-the-art method [168] by large margins. In 5-shot, our method consistently brings a 0.5 $\sim$ 2.7% performance gains to all baselines.

Using representative samples to train our VAE model further improves the three baseline methods under all settings and for all datasets. Compared to SVAE, training on strictly representative data improves the 1-shot classification accuracy by 0.3% ∼ 2.8% and the 5-shot classification accuracy by 0.2% ∼ 0.8%. R-SVAE achieves state-of-the-art few-shot classification on *mini*ImageNet dataset with the ProtoNet baseline and on *tiered*ImageNet dataset with the E3BM baseline.

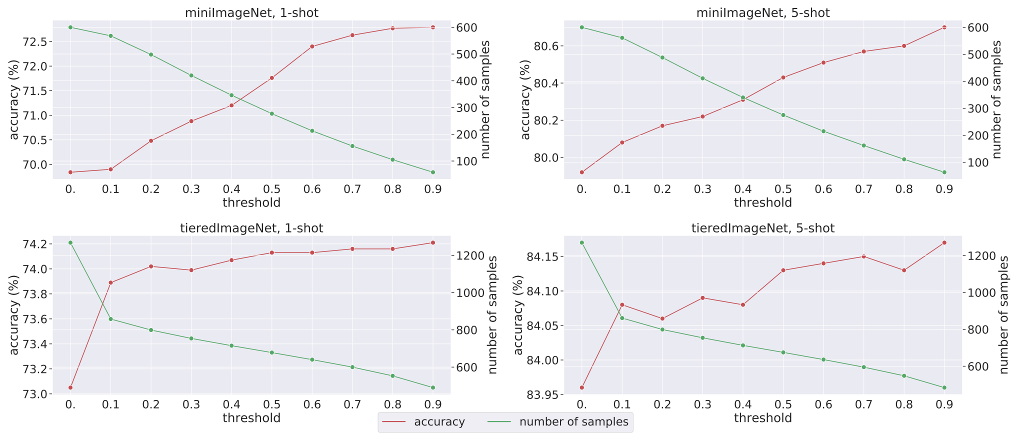


Figure 3.3: **Few-shot classification results with different probability thresholds.** We report the classification accuracy (%) (red) and the number of samples (green) when setting different thresholds for the probabilities. A higher threshold means we select samples that are more representative, resulting in a less amount of training data points. In general, the classification performance increases when the number of training samples decreases with increasing representativeness thresholds.

3.4 Analyses

All the following analyses use the feature extractor from the Meta-Baseline method [24].

3.4.1 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

Classifier	*mini*ImageNet support samples	*mini*ImageNet + SVAE	*mini*ImageNet + R-SVAE	*tiered*ImageNet support samples	*tiered*ImageNet + SVAE	*tiered*ImageNet +R-SVAE
Prototype [24]	63.17 ± 0.23	69.96 ± 0.21	**72.79 ± 0.19**	68.62 ± 0.27	73.05 ± 0.24	**73.90 ± 0.24**
1-N-N	63.28 ± 0.23	67.25 ± 0.20	**69.27 ± 0.19**	68.73 ± 0.26	68.05 ± 0.25	**69.82 ± 0.24**
SVM	63.41 ± 0.23	70.30 ± 0.20	**72.84 ± 0.19**	68.88 ± 0.25	69.26 ± 0.25	**71.28 ± 0.24**
LR	63.33 ± 0.22	72.11 ± 0.20	**73.41 ± 0.19**	69.15 ± 0.25	74.99 ± 0.23	**75.98 ± 0.23**

Table 3.2: **Choices of the classifiers**. One-shot classification accuracy on *mini*ImageNet and *tiered*ImageNet using different types of classifiers, *i.e.*, 1-N-N, SVM and LR. All methods use the feature extractor from the Meta-Baseline method [24].

experiments with different threshold values to see how it affects the classifier's performance. Fig. 3.3 shows the classification accuracy under different thresholds on *mini*ImageNet and *tiered*ImageNet 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 values under both 1-shot and 5-shot settings. For example, under the 1-shot setting on *mini*ImageNet dataset, we only use 58 images per class on average when setting the threshold to 0.9. Training the VAE model with this small set of images improves the performance by 2.95% compared with the model trained using all data in the base set with 600 images per class on average. The results suggest that the performance of our method strongly corresponds to the representativeness of training data. Moreover, it shows that our sample selection method provides a reliable measurement for the representativeness of the training samples.

3.4.2 Performance with Different Classifiers

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 3.2 shows the 1-shot performance of different classifiers using our generated features on *mini*ImageNet 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 *mini*ImageNet 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

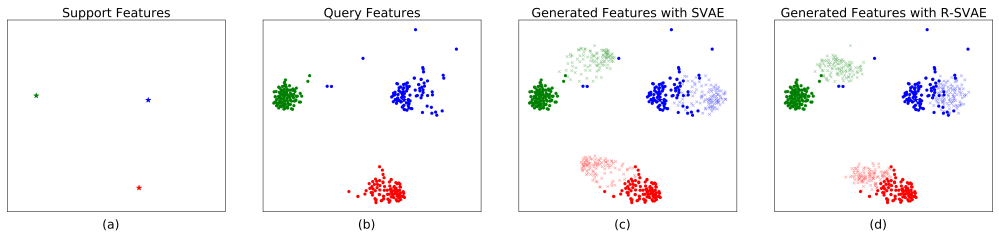


Figure 3.4: **Feature Visualization.** We show the t-SNE visualization of the original features (marked as dark points) and our generated features (marked as transparent points) on *tiered*ImageNet dataset. Different colors represent different classes. From left to right, we show the original support set (a), the query set (b), the features generated by SVAE (c), and the features generated by R-SVAE (d).

can benefit different types of classifiers.

3.4.3 Feature Distribution Analysis

In Fig. 3.4, we show the t-SNE representation [101] 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. 3.4(c) and (d) visualize the effect of our sample selection method. Fig. 3.4(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. 3.4(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:

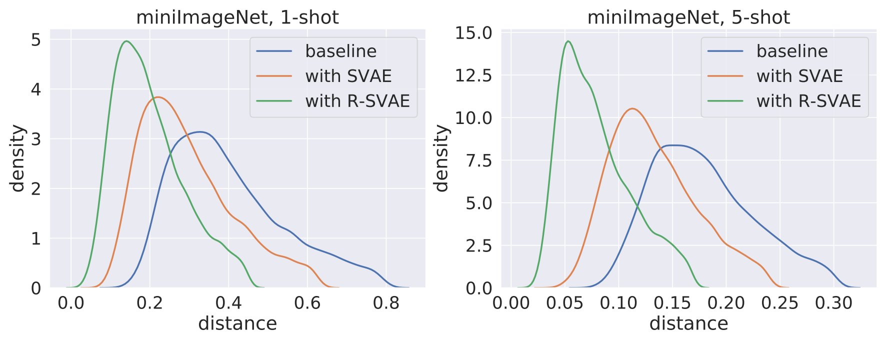


Figure 3.5: **Distance Distributions.** Kernel Density Estimation of the distance between the estimated prototypes and the ground truth prototype. A smaller value means the estimated prototypes are closer to the ground truth prototypes.

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 *mini*ImageNet 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. 3.5. The probability density is calculated by binning and counting observations and then smoothing them with a Gaussian kernel, namely, Kernel Density Estimation [25]. As can be seen the Fig., our estimated class prototypes are much closer to the ground truth prototypes, compared to the baseline.

3.4.4 Sample Visualization

In Fig. 3.6, 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.

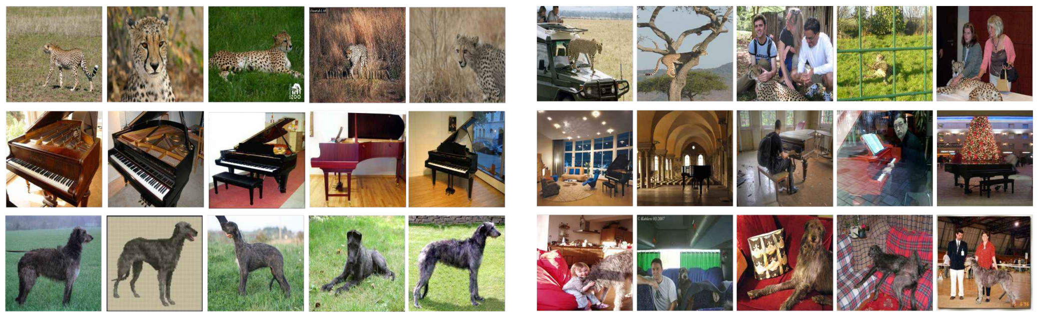


Figure 3.6: **Examples of representative samples (left) and non-representative samples (right)**. We visualize 5 images with high probabilities and 5 images with small probabilities computed via our proposed method for 3 classes from *tiered*ImageNet dataset.

	1-shot	5-shot
Meta-Baseline	63.17 ± 0.23	79.26 ± 0.17
Meta-Baseline + SVAE	67.39 ± 0.21	79.77 ± 0.17
Meta-Baseline + R-SVAE	**68.03** ± **0.22**	**79.93** ± **0.16**

Table 3.3: **Classification accuracy using Word2Vec[106] as the semantic feature extractor.**

3.4.5 Performance with Different Semantic Embedding

We use CLIP features in our main experiments. The performance of our method trained with Word2Vec[106] features are shown in Table 3.3. 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.

3.4.6 Using a Deeper Network for the Decoder

The decoder of our proposed VAE plays a vital role in our framework as it maps the latent space of the VAE and the semantic embedding to the visual feature embedding space. We use a network with two fully-connected (FC) layers for the decoder in our main setting. We experiment with a deeper network where we add an FC layer with 4096 hidden units and a LeakyReLU [55] layer to the decoder. Table 3.4 summarizes the results. Using a deeper network degrades the performance of our model under both 1-shot and 5-shot settings for both *mini*ImageNet [159] and *tiered*ImageNet [131].

	*mini*ImageNet		*tiered*ImageNet	
Decoder	1-shot	5-shot	1-shot	5-shot
2-FC Layers (Main paper)	72.79 ± 0.19	80.70 ± 0.16	74.21 ± 0.24	84.17 ± 0.18
3-FC Layers	71.68 ± 0.20	80.32 ± 0.16	73.84 ± 0.24	84.05 ± 0.25

Table 3.4: **Few-shot classification performance of our method using different network architecture for the decoder**. In our main setting, we use as our decoder a network with two fully-connected (FC) layers. "3-FC Layers" denotes the setting where we add an FC layer with 4096 hidden units and a LeakyReLU layer to the decoder. The performance degrades for both 1-shot and 5-shot settings with a deeper network.

3.4.7 Performance on CIFAR-FS and FC-100

In Table 3.5, we provide the performance of our method on two additional FSL datasets - CIFAR-FS and FC-100. On these both datasets, our method improves the Meta-Baseline method by large margins.

	Dataset	1-shot	5-shot
Meta-Baseline		64.96 ± 0.51	75.85 ± 0.40
Meta-Baseline + SVAE	CIFAR-FS	72.07 ± 0.45	77.18 ± 0.39
Meta-Baseline + R-SVAE		**73.25 ± 0.44**	**78.89 ± 0.37**
Meta-Baseline		41.31 ± 0.42	51.84 ± 0.40
Meta-Baseline + SVAE	FC-100	45.65 ± 0.40	54.37 ± 0.40
Meta-Baseline + R-SVAE		**45.75 ± 0.40**	**54.44 ± 0.40**

Table 3.5: 1-shot and 5-shot classification accuracy on CIFAR-FS and FC-100.

3.5 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 *mini*ImageNet and *tiered*ImageNet 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 [91, 181]. 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.

Chapter 4

Generating Diverse Intra-Class Variance for Fine-Grained Few-Shot Classification

4.1 Overview

In this Chapter, we focus on generating diverse data with diverse intra-class variance for fine-grained few-shot classification. Fine-grained few-shot learning is a more challenging task compared with general few-shot classification task since fine-grained visual data are hard to collect and costly to annotate [68, 158, 167]. Fine-grained datasets often become quite long-tailed and lead to classifiers overfitting to the abundant classes when trained in vanilla settings. Fine-grained few-shot learning methods alleviate this problem since they learn discriminative class features, among visually similar classes, using as few as 5 or 1 training instances.

Augmenting the few-shot classes by generating additional data is a straightforward way to mitigate issues of overfitting in FSL. Nevertheless, generating diverse data reliably remains an open question [75, 147]. The generated samples should contain the class-discriminative features while exhibiting high intra-class diversity. A typical data synthesis approach is generating new samples based on adversarial frameworks [4, 44, 76, 77, 79, 85, 156, 196]. However, these methods suffer from a lack of diversity in the generated samples as adversarial training often mode-collapses. Another approach is the feature transfer that transfers

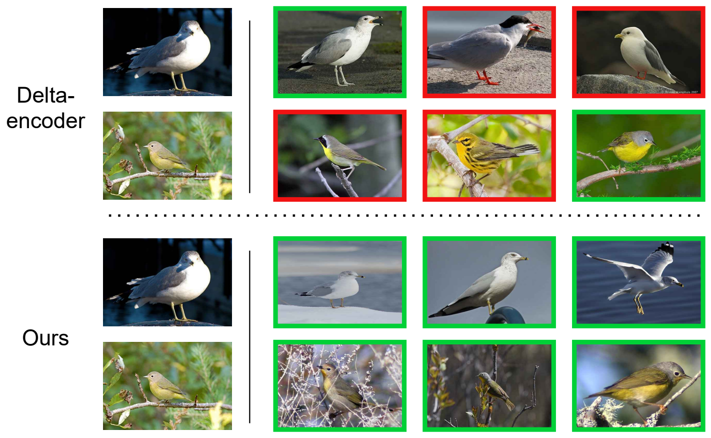


Figure 4.1: **Nearest "real sample" neighbors of the augmented examples.** We train data augmentation methods using the base classes and search for the nearest-neighbors of the generated samples in the novel classes of the CUB dataset. The input images are shown in the first column. Each row shows the nearest neighbors of some augmented features computed from: Δ-encoder [141] (1st and 2nd rows) and our method (3rd and 4th rows). Green borders indicate that the images have the same class as the input image and red borders indicate otherwise.

the intra-class variance from the base classes, which have many training samples, to augment features for the novel classes, in which only few samples are available [53, 141, 188]. These methods are based on a common assumption that intra-class variations induced by poses, backgrounds, or illumination conditions are shared across categories. The intra-class variations are either modelled as low-level statistics [188] or pairwise variations [53, 141] and are applied directly on the novel samples. In this work, we discuss two potential issues with these approaches. First, these transformations can introduce certain class-discriminative features that could alter the class-identity of the transformed features. For example, only 8.7% of the augmented features using the Δ-encoder[141] have their nearest "real sample" neighbors belong to the same classes as the original samples (see Fig. 4.1). Second, the extracted variations might not be relevant to a specific novel sample, i.e., some bird species would never appear in sea backgrounds. Applying irrelevant variations would result in noisy or meaningless samples and degrade classification results (see Sec. 4.4.1). These two issues are more pronounced for fine-grained classification since a small change in feature space might change the category of the feature due to the small inter-class distances.

We address these issues in this work via a novel data augmentation framework. First, we disentangle each feature into two components: one that captures the intra-class variance, which we refer as intra-class variance features, and the other that encodes the class-discriminative features. Second, we model intra-class variance via a common distribution from which we can easily sample the new intra-class variations that are relevant for diversifying a specific instance. We show that both the feature disentanglement and the distribution of intra-class variability can be approximated using data from the base classes and it generalizes well to the novel classes. The two key supervision signals that drive the training of our framework are: 1) A classification loss that ensures that the class-discriminative features contain class specific information, 2) A Variational Auto-Encoder (VAE) [72] system that explicitly models intra-class variance via an isotropic Gaussian distribution. Our method works especially well for fine-grained datasets where the intra-class variations are similar across classes, achieving state-of-the-art few-shot classification performances on the CUB[167], NAB[158], and Stanford Dogs[68] datasets, outperforming previous methods [83, 141] by a large margin. We show in our analyses that the data generated by our method lies closely to the real-and-unseen features of the same class and can closely approximate the distribution of the real data.

To sum up, our contributions are:

1. We are the first to propose a VAE-based feature disentanglement method for fine-grained FSL.

2. We show that we can train such a system using sufficient data from the base classes. We can sample from the learnt distribution to obtain relevant variations to diversify novel training instances in a reliable manner.

3. Our method outperforms state-of-the-art FSL methods in multiple fine-grained datasets by a large margin.

4.2 Proposed Method

4.2.1 Few-shot Learning Preliminaries

In FSL, abundant labeled images of base classes and a small number of labeled images of novel classes are given. Our goal is to train a classifier that can correctly classify novel class images with the few given examples. The standard FSL procedure includes a training stage and a fine-tuning stage. During the training stage, we use base class images to train a feature extractor and the classifier. Then in the fine-tuning stage, we freeze the parameters of the pre-trained feature extractor and train a new classifier head using the few labeled examples in the novel classes . In the testing stage, the learned classifier predicts labels on a set of unseen novel class images.

Since the available samples during the fine-tuning stage are scarce and lack diversity, the learned classifier tends to overfit to the few samples and thus performs poorly on the test images. To address this, we augment the training samples with our proposed data augmentation method, which significantly improves the performance of the baseline.

4.2.2 Overall Pipeline

Our goal is to generate additional features of the few novel class images which contain larger intra-class variance. Fig. 4.2 illustrates the pipeline of our proposed method. We decompose the feature representation of an input image into two components, the class-specifc feature z_I and the intra-class variance feature z_V. z_V is constrained to follow a prior distribution. Then we repeatedly sample new intra-class variance features $\tilde{z}_V$ from the distribution and add them to the class-specific feature z_I to get augmented features. The augmented features are used together with the original features to train the final classifier. In the following sections, we will describe how we model the distribution of intra-class variability via variational inference and how we use it to diversify samples from the novel set.

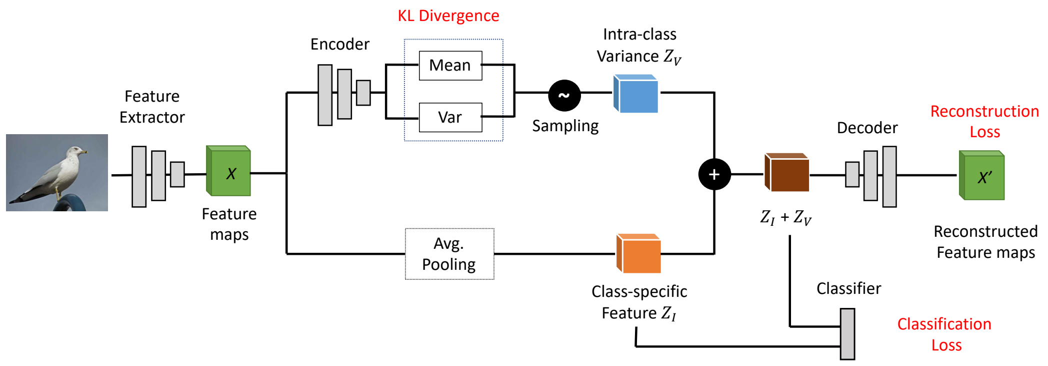


Figure 4.2: **The pipeline of our proposed method**. The input image is mapped into the image feature maps $\boldsymbol{X}$. We input $\boldsymbol{X}$ into an Encoder to obtain the mean and variance of the intra-class variability distribution that are used to sample the intra-class variance feature z_V. The class-specific feature z_I is obtained by max-pooling $\boldsymbol{X}$. z_V is forced to follow an isotropic multivariate Gaussian distribution. Both z_I and the combined features are used to train a classifier. We sample from the learned distribution repeatedly to get multiple z_V and add them to the class-specific feature z_I to get the augmented features. These augmented features are used together with the original ones to train a more robust classifier.

4.2.3 Variational Inference for Intra-class Variance

Given an input image (i), we first use a feature extractor to map it into a feature map $X^{(i)}$. We then compute the intra-class variance feature $z_V^{(i)}$ and the class-specific feature $z_I^{(i)}$ from $X^{(i)}$ such that the embedding of the input image, $z^{(i)}$, can be expressed as:

$$z^{(i)} = z_I^{(i)} + z_V^{(i)}. \tag{4.1}$$

Here we assume that the intra-class variance feature is generated from some conditional distribution $p(z_V)$ and the feature map $X^{(i)}$ is generated from some conditional distribution $p(X|z)$.

The class-specific feature $z_I^{(i)}$ can be learned by minimizing the cross-entropy loss given the class label $y^{(i)}$:

$$L_{cls}(X^{(i)}) = L_{cross-entropy}\left(W(z_I^{(i)}), y^{(i)}\right) \tag{4.2}$$

where W is a classifier with a single fully connected layer.

We use variational inference to model the posterior distribution of the variable z_V. Specifically, we approximate the true posterior distribution $p(z_V|X)$ with another distribution $q(z_V|X)$. The Kullback-Leibler divergence between the true distribution and the approximation is:

$$KL[q(z_V|X)\|p(z_V|X)] = \int_Z q(Z|X)\log\frac{q(Z|X)}{p(Z|X)}. \tag{4.3}$$

Since the Kullback-Leibler divergence is always greater than or equal to zero, maximizing the marginal likelihood $p(X^{(i)})$ is equivalent to maximizing the evidence lower bound (ELBO) defined as follows:

$$\begin{aligned} ELBO^{(i)} = E_{q(z_V^{(i)}|X^{(i)})}[\log p(X^{(i)}|z_V^{(i)})] \\ - KL\Big(q(z_V^{(i)}|X^{(i)})\|p(z_V)\Big). \end{aligned} \tag{4.4}$$

Prior work [91, 188, 194] has shown that the distribution of intra-class variability can be modelled with a Gaussian distribution. Here we set the prior distribution of z_V to be a centered isotropic multivariate Gaussian: $p(z_V) = \mathcal{N}(0, I)$. For the posterior distribution, we set it to be a multivariate Gaussian with diagonal covariance:

$$q(z_V^{(i)}|X^{(i)}) = \mathcal{N}(\mu^{(i)}, \sigma^{(i)}), \tag{4.5}$$

where $\mu^{(i)}$ and $\sigma^{(i)}$ are computed by a probablistic encoder. With the reparameterization trick, we obtain $z_V^{(i)}$ as follows:

$$z_V^{(i)} = \mu^{(i)} + \sigma^{(i)} * \epsilon, \epsilon \sim \mathcal{N}(0, I). \tag{4.6}$$

Since $z_I^{(i)}$ is deterministic given $X^{(i)}$, we have $p(X^{(i)}|z_V^{(i)}) = p(X^{(i)}|z_V^{(i)}, z_I^{(i)}) = p(X^{(i)}|z^{(i)})$. To estimate the maximum likelihood $p(X^{(i)}|z^{(i)})$, we use a decoder to reconstruct the original feature map from $z^{(i)}$ and minimize the $L2$ distance between the original feature map and the reconstructed one.

From Eq. 4.4, we now derive the loss function for the modeling of intra-class variance:

$$L_{intra}(X^{(i)}) = \|X^{(i)} - \hat{X}^{(i)}\|^2 + KL\Big(q(z_V^{(i)}|X^{(i)})\|p(z_V)\Big), \tag{4.7}$$

where $\hat{X}^{(i)}$ is the reconstructed feature map synthesized from the sum of class-specific feature $z_I^{(i)}$ and intra-class variance feature $z_V^{(i)}$ sampled from the distribution $\mathcal{N}(\mu^{(i)}, \sigma^{(i)})$.

The L_{intra} loss includes two terms. The first term is the reconstruction term, which ensures that the encoder extracts meaningful information from the inputs. The second term is a regularization term, which forces the latent code, $z_V^{(i)}$, to follow a standard normal distribution. Here, instead of minimizing the Kullback-Leibler divergence directly, we decompose it into three terms as in [21]:

$$\begin{aligned} KL[q(z_V|X)||p(z_V)] &= KL(q(z_V, X)||q(z_V)p(X)) + \\ & KL(q(z_V)||\prod_j q(z_{V_j})) + \sum_j KL(q(z_{V_j})||p(z_{V_j})), \end{aligned} \tag{4.8}$$

where z_{V_j} denotes the j-th dimension of the latent variable.

The three terms in Eq. 4.8 are referred to as the *index-code mutual information*, *total correlation*, and *dimension-wise* KL respectively. Prior work [3, 17, 21] has shown that penalizing the index-code mutual information and total correlation terms leads to a more disentangled representation while the dimension-wise KL term ensures that the latent variables do not deviate too far form the prior. Similar to [21], we penalize the total correlation with a weight α and rewrite L_{intra} as follows:

$$\begin{aligned} L_{intra}(X^{(i)}) &= ||X^{(i)} - \hat{X}^{(i)}||^2 + KL\left(q(z_V^{(i)}, X^{(i)})||q(z_V^{(i)})p(X)\right) + \\ & \alpha * KL\left(q(z_V^{(i)})||\prod_j q(z_{V_j}^{(i)})\right) + \sum_j KL\left(q(z_{V_j}^{(i)})||p(z_{V_j})\right). \end{aligned} \tag{4.9}$$

The combination of L_{cls} and L_{intra} drives the model to extract discriminative class-specific features $z_I^{(i)}$ and model the distribution of intra-class variability simultaneously.

4.2.4 Objective Function

Given the distribution of intra-class variability, we can generate additional samples for the base classes during the training stage. For input image (i)

with extracted class-specific feature $z_I^{(i)}$ and intra-class variability mean and variance $\mu^{(i)}$ and $\sigma^{(i)}$ respectively, we sample new intra-class variance features, $\tilde{z}_V^{(i)}$, for this image from the distribution $\mathcal{N}(\mu^{(i)}, \sigma^{(i)})$ and add them to $z_I^{(i)}$ to obtain the augmented features $\tilde{z}^{(i)} = z_I^{(i)} + \tilde{z}_V^{(i)}$. We use these features to train our system using the following cross-entropy loss:

$$L_{aug}(X^{(i)}) = L_{cross-entropy}\Big(W(\tilde{z}^{(i)}), y^{(i)}\Big). \tag{4.10}$$

The overall loss function in the training stage is a weighted combination of the aforementioned terms:

$$L = L_{cls} + L_{intra} + \beta * L_{aug}, \tag{4.11}$$

where β is the coefficient of L_{aug}.

4.2.5 Diversifying Samples for Few-Shot Classes

In this section, we discuss how to use our model to diversify samples for few-shot classes. Our intra-class variance is modelled by an isotropic Gaussian distribution. Sampling from this distribution would result in an arbitrary intra-class variance feature. However, we conjecture that such an arbitrary feature may not be relevant for all instances, i.e., some birds never appear with a background of the sea. Note that here as all intra-class variations are mapped into a common continuous embedding space via variational inference and closely related or similar intra-class variations likely form local neighborhoods in the embedding space. Thus, instead of sampling from the zero-mean and unit-variance distribution, we only sample from the mean and variance estimated directly from the conditional sample to obtain the likely relevant intra-class variations to this sample.

Specifically, given an image of novel class $(i)^*$ with class label $y^{(i)*}$, we first extract the feature map $X^{(i)*}$, the class-specific feature $z_I^{(i)*}$, and the mean and variance of the intra-class variability distribution $\mu^{(i)*}$ and $\sigma^{(i)*}$ for this instance. We then generate additional features by adding the class-specific features $z_I^{(i)*}$ with a biased term sampled from the distribution of intra-class variability.

$$\tilde{z}^{(i)*} = z_I^{(i)*} + \tilde{z}_V^{(i)*}, \tilde{z}_V^{(i)*} \sim N(\mu^{(i)*}, \sigma^{(i)*}), \tag{4.12}$$

where $\tilde{z}^{(i)*}$ is the augmented feature and $\tilde{z}_V^{(i)*}$ is sampled from the posterior distribution $N\left(\mu^{(i)*}, \sigma^{(i)*}\right)$. By sampling from $N\left(\mu^{(i)*}, \sigma^{(i)*}\right)$ multiple times, we get multiple augmented features $\tilde{z}^{(i)*}$ that can be used to train the classifier. In Sec. 4.4.1, we verify the effectiveness of this sampling scheme.

4.3 Experiments

4.3.1 Datasets

We evaluate our method on three fine-grained image classification datasets: Caltech UCSD Birds (CUB) [167], North America Birds (NAB) [158] and Stanford Dogs [68]. The CUB dataset contains 11,788 bird images from 200 bird species in total. Following the setup introduced in [167], we sample the base classes from the 100 classes provided for training, and sample the novel set from the 50 classes provided for testing. The NAB dataset contains 48,527 bird images with 555 classes, which is four times larger than CUB. Similar to [156], we adopt a 2:1:1 training, validation and test set split. The Stanford Dogs dataset is a subset of the Imagenet dataset designed for fine-grained image classification with 90 categories for training and validation and 30 testing categories.

4.3.2 Implementation Details

We conduct experiments with two architectures of our feature extractor: ResNet12 and Conv4 for fair comparisons with other methods using similar architectures. **ResNet12** [56] contains 4 Residual blocks. Each residual block is composed of 3 *conv* layers with 3×3 kernels. A 2×2 max-pooling layer is applied at the end of each residual block. **Conv4** consists of 4 layers with 3×3 convolutions and 32 filters, followed by batch normalization (BN) , a ReLU nonlinearity, and 2×2 max-pooling.

The class-specific features are calculated by average-pooling the output of the feature extractor. The encoder consists of three *conv* blocks followed by two fully-connected heads that output the μ and $\log\sigma^2$ respectively. The decoder consists of a fully connected layer followed by three Convolutional blocks.

Method	CUB		NAB		Stanford Dogs	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Baseline [23]	63.90 ± 0.88	82.54 ± 0.54	70.36 ± 0.89	87.91 ± 0.49	63.53 ± 0.89	79.95 ± 0.59
Baseline++ [23]	68.46 ± 0.85	81.02 ± 0.46	76.00 ± 0.85	90.99 ± 0.41	58.30 ± 0.35	73.77 ± 0.68
MAML [42]	71.11 ± 1.00	82.08 ± 0.72	80.08 ± 0.93	88.87 ± 0.54	66.56 ± 0.66	79.32 ± 0.35
MatchingNet [159]	72.62 ± 0.90	84.14 ± 0.50	73.91 ± 0.72	88.17 ± 0.45	65.87 ± 0.81	80.70 ± 0.42
ProtoNet [150]	71.57 ± 0.89	86.37 ± 0.49	73.60 ± 0.83	89.72 ± 0.41	65.02 ±0.92	83.69 ± 0.48
RelationNet [154]	70.20 ± 0.84	84.28 ± 0.46	67.41 ± 0.82	85.47 ± 0.43	59.38 ± 0.79	79.10 ± 0.37
MTL [153]	73.31 ± 0.92	82.29 ± 0.51	78.69 ± 0.78	87.74 ± 0.34	54.96 ± 1.03	68.76 ± 0.65
Δ-encoder [141]	73.91 ± 0.87	85.60 ± 0.62	79.42 ± 0.77	92.32 ± 0.59	68.59 ± 0.53	78.60 ± 0.78
MetaOptNet [83]	75.15 ± 0.46	87.09 ± 0.30	84.56 ± 0.46	93.31 ± 0.22	65.48 ± 0.49	79.39 ± 0.25
Ours	**79.12** ± 0.83	**91.48** ± 0.39	**88.62** ± 0.73	**95.22** ± 0.32	**76.24** ± 0.87	**88.00** ± 0.47

Table 4.1: Few-shot classification accuracy on the CUB [167], NAB [158], and Stanford Dogs [68] dataset. All experiments are from 5-way classification with the same backbone network (ResNet12). The best performance is indicated in bold.

Training policies. The whole network is trained from scratch in an end-to-end manner. In the training stage, we use the Adam optimizer [70] on all datasets with initial learning rate 0.001 . We train our model for 100 epochs in total with a batch size of 16 and reduce the learning rate by 0.1 at the 40-th and 80-th epochs. We empirically set $\alpha = 4$ in Eq. 4.9 and $\beta = 1$ in Eq.4.11.

We follow a standard few-shot evaluation scheme. In the fine-tuning stage, we select 5 classes from the novel classes randomly. For each class, we pick k instances as the support set and 16 instances for the query set for a k-shot task. The extracted features of all support set images along with the augmented features are used to train a linear classifier for 100 iterations with a batch size of 4. For each feature extracted from a support image, we obtain five augmented features. The final results are averaged over 600 experiments. For data augmentation, we adopt random cropping, horizontal flipping and color jittering as in [23]. The final size of the input images is 84×84 .

4.3.3 Results

Tab. 4.1 summarizes the 5-way classification accuracy of various methods using ResNet12 backbones. The results are obtained using the publicly available code of each method. Our proposed method outperforms the previous methods by a large margin for both 1-shot and 5-shot settings on all three datasets. Compared with the Δ-encoder [141], another data aug-

Method	CUB		Stanford Dogs	
	1-shot	5-shot	1-shot	5-shot
MatchingNet [159]	45.30 ± 1.03	59.50 ± 1.01	35.80 ± 0.99	47.50 ± 1.03
ProtoNet [150]	37.36 ± 1.00	45.28 ± 1.03	37.59 ±1.00	48.19 ± 1.03
RelationNet [154]	58.99 ± 0.52	71.20 ± 0.40	43.29 ± 0.46	55.15 ± 0.39
MAML [42]	58.13 ± 0.36	71.51 ± 0.30	44.84 ± 0.31	58.61 ± 0.30
adaCNN [111]	56.76 ± 0.50	61.05 ± 0.44	42.16 ± 0.43	54.12 ± 0.39
CovaMNet [98]	52.42 ± 0.76	63.76 ± 0.64	49.10 ± 0.76	63.04 ± 0.65
DN4 [86]	53.15 ± 0.84	81.90 ± 0.60	45.73 ± 0.76	61.51 ± 0.85
LRPABN [61]	63.63 ± 0.77	76.06 ± 0.58	45.72 ± 0.75	60.94 ± 0.66
MattML [203]	66.29 ± 0.56	80.34 ± 0.30	54.84 ± 0.53	71.34 ± 0.38
Ours	**68.42** ± 0.92	**82.42** ± 0.61	**57.03** ± 0.86	**73.00** ± 0.66

Table 4.2: Few-shot classification accuracy on the CUB [167] and Stanford Dogs [68] dataset. All experiments are from 5-way classification with the same backbone network (Conv4). The best performance is indicated in bold.

mentation based method, our proposed method achieves 7.40%, 9.20% and 7.65% performance gain for the 1-shot setting and 5.88%, 2.90% and 9.40% performance gain for the 5-shot setting on the three datasets respectively. It can be seen that our improvement in the 1-shot setting is more pronounced than in the 5-shot setting since the 1-shot setting is a more extreme case of data scarcity, in which augmenting the training data tends to be more useful.

We compare with methods using Conv4 architectures as the backbone networks in Tab. 4.2. Here the majority of methods only report their results on the CUB and Stanford Dogs datasets. Our proposed method achieves state-of-the-art performance for both the 1-shot and 5-shot settings. Especially for the 1-shot setting, our method obtains 2.12% performance gain for the CUB and 2.19% gain for the Stanford Dogs over MattML [203], a newly proposed method that is aimed specifically at fine-grained few-shot visual recognition.

Our method also achieves competitive few-shot classification performances on non fine-grained datasets such as CIFAR-FS[11] and mini-ImageNet[129, 159].

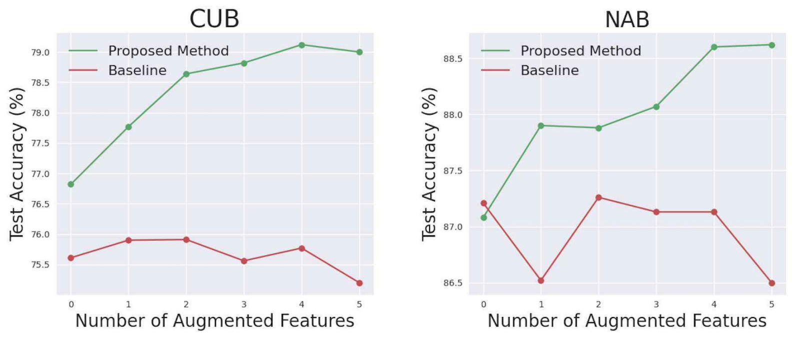


Figure 4.3: **Analysis on the generated intra-class variations.** We augment samples with the intra-class variance features sampled from the estimated mean and variance (green lines) or from the zero-mean and unit-variance (red lines). Our sampling scheme generates features that consistently improve classification.

4.4 Analyses

In this section, we provide additional experiments to clarify different aspects of our methods.

4.4.1 Analysis on the Generated Intra-class Variations

We conduct a simple experiment to verify the effectiveness of our sampling method (Sec.4.2.5). Instead of sampling from the instance-conditioned mean and variance, we sample the intra-class variance feature from the zero-mean and unit-variance distribution.

Fig. 4.3 summarizes the results of this experiment for 5-way 1-shot classification on the CUB and NAB dataset. As can be seen, intra-class variance features sampled from zero-mean and unit-variance do not improve the results (red lines). In contrast, our method of sampling from the instance-conditioned posterior distribution generates features that consistently improve classification performance as the number of augmented samples increases.

4.4.2 Comparison to Other Augmentation Methods

We compare our method with two other data augmentation based FSL methods: MetaIRNet[156] and Δ-encoder[141]. MetaIRNet uses a pre-trained image generator to synthesize additional images and combine them with the original images to form additional training samples. The Δ-encoder learns to synthesize transferable non-linear deformations between pairs of examples of seen classes and apply these deformations to the few provided samples of novel categories.

We use the additional samples synthesized by both of these methods to train three types of classifiers: K-nearest neighbors (KNN), Support Vector Machine (SVM), and Logistic Regression (LR), which are then used to classify novel images. The comparisons between these methods and our method are shown in Tab. 4.3. The superior performance of our method demonstrates that the augmented features obtained by our framework is beneficial for various types of classifiers. Note that for MetaIRNet [156], the results in Tab. 4.3 are lower than their numbers reported in the original paper since they pre-trained the backbone network on ImageNet while here all methods are trained from scratch.

Method	KNN		SVM		LR	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
MetaIRNet [156]	63.18	74.82	63.76	76.77	63.53	79.95
Δ-Encoder [141]	67.31	82.67	76.02	82.87	76.22	85.17
Ours	**75.46**	**83.17**	**79.07**	**87.59**	**78.34**	**89.30**

Table 4.3: **Analysis of different classifiers.** Few-shot classification accuracy on the CUB [167] dataset in 1-shot and 5-shot settings with different types of classifiers.

In Tab. 4.4, we directly compare our method withthe Δ-encoder using K-NN classifiers (K=1). Interestingly, it can be seen that the augmented features generated using the delta-encoder decrease classification performance. In fact, we observe that the majority (91.3%) of the nearest neighbors of the Δ-encoder's generated features belong to different classes, suggesting that the pairwise transformations extracted from this method might alter the class-identities of the transformed features. On the other hand, our generated features preserve well the class identity and mildly improve the classification results.

Method	Δ-Encoder		Ours	
	w/o Aug	w/ Aug	w/o Aug	w/ Aug
5-way	69.37	67.31	74.95	75.46
10-way	58.69	52.19	62.05	63.17
20-way	48.10	38.84	50.19	50.72

Table 4.4: **Effect of augmented features on 1NN classifier.** Few-shot classification accuracy on the CUB [167] dataset using 1NN classifier with original features vs augmented features. The original features of the Δ-Encoder are from a pre-trained ResNet18 network.

Intra-class distribution model	CUB		NAB	
	1-shot	5-shot	1-shot	5-shot
Gaussian Mixture Model [33]	75.16	86.46	86.49	94.72
Covariance Matrix [188]	75.28	87.84	84.71	94.19
No disentanglement [194]	73.40	86.60	81.83	92.83
Isotropic Gaussian (Proposed)	**79.12**	**91.48**	**88.62**	**95.22**

Table 4.5: Few-shot classification accuracy on the CUB [167] and NAB [158] dataset in 1-shot and 5-shot setting with different methods to model intra-class variance.

4.4.3 Comparison to Other Intra-class Variance Modeling Methods

We assume that the intra-class variance can be modelled with an isotropic multivariate Gaussian distribution in a latent space. In this section, we compare this method with other methods that model the intra-class variance including Gaussian mixture variational autoencoder (GMVAE) [33], covariance matrix [188], and a baseline model where we do not disentangle intra-class variance features from class-discriminative features.

Tab. 4.5 summarizes the results. The first row shows the results for GMVAE. This method enforces that the latent space is divided into distinct clusters for different classes. However, for this model, the accuracy drops by 6.15% and 2.13% for the 1-shot setting and 5.02% and 0.50% for the 5-shot setting on the CUB and NAB datasets respectively. The results align with our assumption that the intra-class variance is shared across different classes. Thus, enforcing a multi-modal prior distribution would lead to performance degradation.

The second row shows the results for the method proposed in [188]

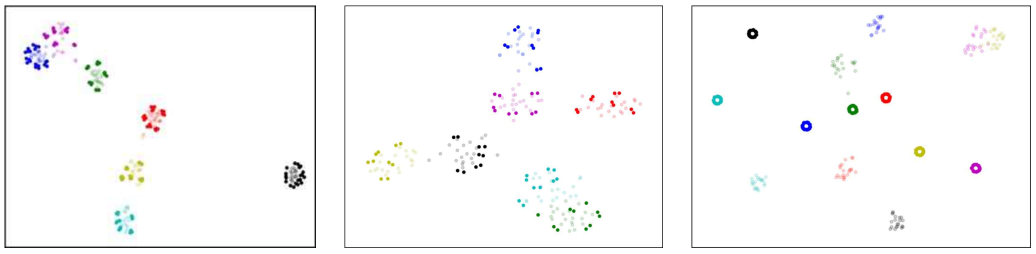

(a) Ours - Base Classes (b) Ours - Novel Classes (c) Δ-encoder - Novel Classes

Figure 4.4: **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.

based on covariance matrices. Specifically, this method assumes a Gaussian prior on the distribution of intra-class variability across different classes which can be transferred from the base classes to the rare classes. However, instead of modelling the distribution by variational inference, [188] uses a covariance matrix to estimate the feature variance distribution.Here we apply this method on our extracted features to generate additional features on the CUB and NAB datasets under both 1-shot and 5-shot settings. Compared with the non-parametric estimate of the Gaussian distribution, modelling intra-class variance via variational inference in an end-to-end manner brings 6.03% and 1.91% improvement for the 1-shot setting and 3.64% and 1.03% improvement for the 5-shot setting on the CUB and NAB dataset respectively. Last, we provide the results for our method without feature disentanglement, denoted as "No disentanglement" in the third row. In spirit, this model is similar to [194] which models each point as a distribution via variational inference. Given a new sample, we augment it via sampling repeatedly from the estimated mean and variance. Without feature disentanglement and explicit modelling of the intra-class variance, this model does not achieve comparable results compared to other methods.

4.4.4 Data Distribution Analysis

We compare the data distributions between the real data and the generated data from our method in comparison to other state-of-the-art data generation methods [141, 188]. Here we measure the average intra-class variance, the distances between classes (inter-class distances), and the data clusterability via the Davies–Bouldin index (DBI) [28]. Specifically, the DBI for a cluster i is calculated by:

$$DBI_i = \max_{i \neq j} \frac{Intra_{(i)} + Intra_{(j)}}{Inter_{(i,j)}} \tag{4.13}$$

where $Intra_{(i)}$ is the intra-class variance of cluster i, calculated by taking the average of squared deviations from the class center. $Inter_{(i,j)}$ is the distance between the two class centers of clusters i and j. The lower the value of the DBI, the better the separation between the clusters and the "tightness" inside the clusters.

	D_{intra}	D_{inter}	DBI
Support data (5 samples)	21.52	32.77	2.21
All data	28.97	35.89	3.02
Covariance matrix [188]	17.98	35.24	1.79
Encoder-based Model [141]	10.34	11.69	1.77
Ours	27.27	34.12	2.53

Table 4.6: **Data Distribution analysis for different sets of features**. We augment features using our method and other data generation method based on covariance matrices [188] or the Δ-encoder [141]. All methods augment features from the support set (first row).

Tab. 4.6 shows the average values of the intra-class variance, inter-class distances, and the DBI (denoted as D_{intra}, D_{inter}, and DBI respectively) across all novel classes of the CUB dataset. The inter-class distances are averaged across all pairs of classes. As shown in the table, features from the support set exhibit smaller intra-class variance compared to features from all data. All methods augment features from the support set. Interestingly, both sets of generated features using the method proposed in [188] and the Δ-encoder[141] decrease intra-class variance. On the other hand, the set of features augmented by our method closely approximate the data distribution of the set of all real features.

Fig. 4.4 demonstrates how real samples and generated samples from our method are distributed in a 2D space in comparison with the Δ-encoder [141] using t-SNE [101]. The original features are marked as light colors while the augmented features are marked as dark colors. Different colors denote different classes. The visualization for the base classes with the augmented features from our method is shown in Fig. 4.4a. Fig. 4.4b visualizes the real features and the generated features of our method for the novel classes. Our method generates samples that follow closely the real samples. The visualization for the novel classes and the generated features from the Δ-encoder is shown in Fig. 4.4c. As can be seen, the generated data from each novel class forms into a new cluster and does not lie close to the actual data points.

4.5 Conclusion

We have proposed a simple, yet effective, feature augmentation method via feature disentanglement and variational inference to address the data scarcity problem in few-shot fine-grained classification. The generated features enlarge the intra-class variance for novel set images while preserving the class-discriminative features. The consistent performance improvement with the increase of the number of augmented samples suggests that the learned features are meaningful and nontrivial. The higher accuracy compared with other data augmentation based methods further demonstrate the superiority of our method. While this work mainly focuses on few-shot recognition problems, a promising future direction is to apply the feature transfer idea to other data-scarce or label-scarce tasks.

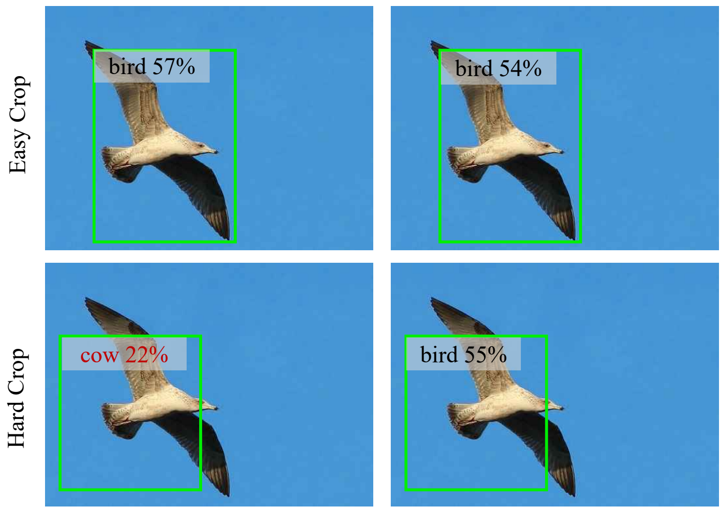
Chapter 5

Generating Data with Increased Crop-Related Diversity for Few-Shot Object Detection

5.1 Overview

In this Chapter, we focus on generating data with increased diversity for few-shot object detection. Object detection plays a vital role in many computer vision systems. However, training a robust object detector often requires a large amount of training data with accurate bounding box annotations. Thus, there has been increasing attention on few-shot object detection (FSOD), which learns to detect novel object categories from just a few annotated training samples. It is particularly useful for problems where annotated data can be hard and costly to obtain such as rare medical conditions [117, 161], rare animal species [167], satellite images [14, 80], or failure cases in autonomous driving systems [102, 103, 133].

For the most part, state-of-the-art FSOD methods are built on top of a two-stage framework [132], which includes a region proposal network that generates multiple image crops from the input image and a classifier that labels these proposals. While the region proposal network generalizes well to novel classes, the classifier is more error-prone due to the lack of training data diversity [152]. To mitigate this issue, a natural approach is to generate additional features for novel classes [54, 197, 202]. For example, Zhang *et al* [197] propose a feature hallucination network to use the variation from base classes to diversify training data for novel classes. For zero-shot detection (ZSD), Zhu *et al* [202] propose to synthesize visual features for unseen objects based on a conditional variational auto-encoder. Although much progress has been made, the lack of data diversity is still a challenging issue for FSOD methods.



(a) DeFRCN [122] (b) Ours

Figure 5.1: **Robustness to different object crops of the same object instance**. (a) The classifier head of the state-of-the-art FSOD method [122] classifies correctly a simple crop of the bird but misclassifies a hard crop where some parts are missing. (b) Our method can handle this case since it is trained with additional generated features with increased crop-related diversity. We show the class with the highest confidence score.

Here we discuss a specific type of data diversity that greatly affects the accuracy of FSOD algorithms. Specifically, given a test image, the classifier needs to accurately classify multiple object proposals[1] that overlap the object instance in various ways. The features of these image crops exhibit great variability induced by different object scales, object parts included in the crops, object positions within the crops, and backgrounds. We observe a typical scenario where the state-of-the-art FSOD method, DeFRCN [122], only classifies correctly a few among many proposals overlapping

[1]Note that an RPN typically outputs 1000 object proposals per image.

an object instance of a few-shot class. In fact, different ways of cropping an object can result in features with various difficulty levels. An example is shown in Figure 5.1 a where the image crop shown in the top row is classified correctly while another crop shown in the bottom row confuses the classifier due to some missing object parts. In general, the performance of the method on those hard cases is significantly worse than on easy cases (see section 5.4.4). However, building a classifier robust against crop-related variation is challenging since there are only a few images per few-shot class.

In this work, we propose a novel data generation method to mitigate this issue. Our goal is to generate features with diverse crop-related variations for the few-shot classes and use them as additional training data to train the classifier. Specifically, we aim to obtain a diverse set of features whose difficulty levels vary from easy to hard *w.r.t.* how the object is cropped.[2] To achieve this goal, we design our generative model such that it allows us to control the difficulty levels of the generated samples. Given a model that generates features from a latent space, our main idea is to enforce that the magnitude of the latent code linearly correlates with the difficulty level of the generated feature, *i.e.*, the latent code of a harder feature is placed further away from the origin and vice versa. In this way, we can control the difficulty level by simply changing the norm of the corresponding latent code.

In particular, we use conditional VAE to build our data generation model. The VAE consists of an encoder that maps the input to a latent representation and a decoder that reconstructs the input from this latent code. In our case, inputs to the VAE are object proposal features, extracted from a pre-trained object detector. The goal is to associate the norm (magnitude) of the latent code with the difficulty level of the object proposal. To do so, we rescale the latent code such that its norm linearly correlates with the Intersection-Over-Union (IoU) score of the input object proposal *w.r.t.* the ground-truth object box. This IoU score is a proxy that partially indicates the difficulty level: A high IoU score indicates that the object proposal significantly overlaps with the object instance while a low IoU score indicates a harder case where a part of the object can be missing. With this rescaling step, we can bias the decoder to generate harder samples by increasing the latent code magnitude and vice versa. In this work,

[2]In this work, the difficulty level is strictly related to how the object is cropped.

we use latent codes with different norms varying from small to large to obtain a diverse set of features which can then serve as additional training data for the few-shot classifier.

To apply our model to FSOD, we first train our VAE model using abundant data from the base classes. The VAE is conditioned on the semantic code of the input instance category. After the VAE model is trained, we use the semantic embedding of the few-shot class as the conditional code to synthesize new features for the corresponding class. In our experiments, we use our generated samples to fine-tune the baseline few-shot object detector - DeFRCN [122]. Surprisingly, a vanilla conditional VAE model trained with only ground-truth box features brings a 3.7% nAP50 improvement over the DeFRCN baseline in the 1-shot setting of the PASCAL VOC dataset [37]. Note that we are the first FSOD method using VAE-generated features to support the training of the classifier. Our proposed Norm-VAE can further improve this new state-of-the-art by another 2.1%, *i.e.*, from 60% to 62.1%. In general, the generated features from Norm-VAE consistently improve the state-of-the-art few-shot object detector [122] for both PASCAL VOC and MS COCO [90] datasets.

The main contributions in this Chapter can be summarized as follows:

- We show that lack of crop-related diversity in training data of novel classes is a crucial problem for FSOD.
- We propose Norm-VAE, a novel VAE architecture that can effectively increase crop-related diversity in difficulty levels into the generated samples to support the training of FSOD classifiers.
- Our experiments show that the object detectors trained with our additional features achieve state-of-the-art FSOD in both PASCAL VOC and MS COCO datasets.

5.2 Proposed Method

In this section, we first review the problem setting of few-shot object detection and the conventional two-stage fine-tuning framework. Then we introduce our method that tackles few-shot object detection via generating features with increased crop-related diversity.

5.2.1 Preliminaries

In few-shot object detection, the training set is divided into a base set D^B with abundant annotated instances of classes C^B, and a novel set D^N with few-shot data of classes C^N, where C^B and C^N are non-overlapping. For a sample $(x, y) \in D^B \cup D^N$, x is the input image and $y = \{(c_i, b_i), i = 1, ..., n\}$ denotes the categories $c \in C^B \cup C^N$ and bounding box coordinates b of the n object instances in the image x. The number of objects for each class in C^N is K for K-shot detection. We aim to obtain a few-shot detection model with the ability to detect objects in the test set with classes in $C^B \cup C^N$.

Recently, two-stage fine-tuning methods have shown great potential in improving few-shot detection. In these two-stage detection frameworks, a Region Proposal Network (RPN) takes the output feature maps from a backbone feature extractor as inputs and generates region proposals. A Region-of-Interest (RoI) head feature extractor first pools the region proposals to a fixed size and then encodes them as vector embeddings, known as the RoI features. A classifier is trained on top of the RoI features to classify the categories of the region proposals.

The fine-tuning often follows a simple two-stage training pipeline, *i.e.*, the data-abundant base training stage and the novel fine-tuning stage. In the base training stage, the model collects transferable knowledge across a large base set with sufficient annotated data. Then in the fine-tuning stage, it performs quick adaptation on the novel classes with limited data. Our method aims to generate features with diverse crop-related variations to enrich the training data for the classifier head during the fine-tuning stage. In our experiments, we show that our generated features significantly improve the performance of DeFRCN [122].

5.2.2 Overall Pipeline

Figure 5.2 summarizes the main idea of our proposed VAE model. For each input object crop, we first use a pre-trained object detector to obtain its RoI feature. The encoder takes as input the RoI feature and the semantic embedding of the input class to output a latent code z. We then transform z such that its norm linearly correlates with the IoU score of the input object crop *w.r.t.* the ground-truth box. The new norm is the output of a simple linear function $g(\cdot)$ taking the IoU score as the single input. The decoder takes as input the new latent code and the class semantic em-

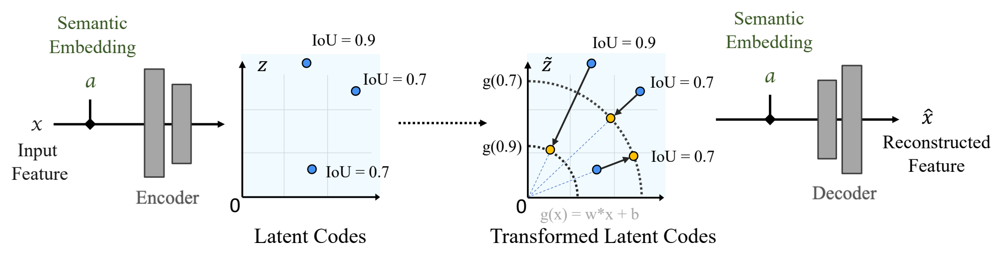


Figure 5.2: **Norm-VAE for modelling crop-related variations.** The original latent code z is rescaled to $\hat{z}$ such that the norm of $\hat{z}$ linearly correlates with the IoU score of the input crop (*w.r.t.* the ground truth box). The original latent codes are colored in **blue** while the rescaled ones are colored in **yellow**. The norm of the new latent code is the output of a simple linear function $g(\cdot)$ taking the IoU score as the single input. As can be seen, the two points whose IoU = 0.7 are both rescaled to norm $g(0.7)$ while another point whose IoU = 0.9 is mapped to norm $g(0.9)$. As a result, different latent norms represent different crop-related variations, enabling diverse feature generation.

bedding to output the reconstructed feature. Once the VAE is trained, we use the semantic embedding of the few-shot class as the conditional code to synthesize new features for the class. To ensure the diversity *w.r.t.* object crop in generated samples, we vary the norm of the latent code when generating features. The generated features are then used together with the few-shot samples to fine-tune the object detector.

Norm-VAE for Feature Generation

We develop our feature generator based on a conditional VAE architecture [151]. Given an input object crop, we first obtain its Region-of-Interest (RoI) feature f via a pre-trained object detector. The RoI feature f is the input for the VAE. The VAE is composed of an Encoder $E(f,a)$, which maps a visual feature f to a latent code z, and a decoder $G(z,a)$ which reconstructs the feature f from z. Both E and G are conditioned on the class semantic embedding a. We obtain this class semantic embedding a by inputting the class name into a semantic model [108, 124]. It contains class-specific information and serves as a controller to determine the categories of the generated samples. Conditioning on these semantic embeddings allows reliably generating features for the novel classes based on the learned information from the base classes [180]. Here we assume that the class names of both base and novel classes are available and we can obtain the semantic embedding of all classes.

We first start from a vanilla conditional VAE model. The loss function for training this VAE for a feature f_i of class j can be defined as:

$$\begin{aligned} L_V(f_i) = \mathrm{KL}\Big(q(z_i|f_i, a^j)||p(z|a^j)\Big) - \\ \mathrm{E}_{q(z_i|f_i, a^j)}[\log p(f_i|z_i, a^j)], \end{aligned} \tag{5.1}$$

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

The goal is to control the crop-related variation in a generated sample. Thus, we establish a direct correspondence between the latent norm and the crop-related variation. To accomplish this, we transform the latent code such that its norm correlates with the IoU score of the input crop. Given an input RoI feature f_i of a region with an IoU score s_i, we first input this RoI feature to the encoder to obtain its latent code z_i. We then transform z_i to $\tilde{z_i}$ such that the norm of $\tilde{z_i}$ correlates to s_i. The new latent code $\tilde{z_i}$ is the output of the transformation function $\mathcal{T}(\cdot, \cdot)$:

$$\tilde{z_i} = \mathcal{T}(z_i, s_i) = \frac{z_i}{||z_i||} * g(s_i), \tag{5.2}$$

where $||z_i||$ is the L_2 norm of z_i, s_i is the IoU score of the input proposal *w.r.t.* its ground-truth object box, and $g(\cdot)$ is a simple pre-defined linear function that maps an IoU score to a norm value. With this new transformation step, the loss function of the VAE from equation 5.1 for an input feature f_i from class j with an IoU score s_i thus can be rewritten as:

$$\begin{aligned} L_V(f_i, s_i) = \mathrm{KL}\Big(q(z_i|f_i, a^j)||p(z|a^j)\Big) - \\ \mathrm{E}_{q\left(z_i|f_i, a^j\right)}\Big[\log p(f_i|\mathcal{T}(z_i, s_i), a^j)\Big]. \end{aligned} \tag{5.3}$$

Generating Diverse Data for Improving Few-shot Object Detection

After the VAE is trained on the base set, we generate a set of features with the trained decoder. Given a class y with a semantic vector a^y and a noise vector z, we generate a set of augmented features $\mathbb{G}^y$:

$$\mathbb{G}^y = \{\hat{f} | \hat{f} = G(\frac{z}{||z||} * \beta, a^y)\}, \tag{5.4}$$

where we vary β to obtain generated features with more crop-related variations. The value range of β is chosen based on the mapping function $g(\cdot)$. The augmented features are used together with the few-shot samples to fine-tune the object detector. We fine-tune the whole system using an additional classification loss computed on the generated features together with the original losses computed on real images. This is much simpler than the previous method of [197] where they fine-tune their system via an EM-like (expectation-maximization) manner.

5.3 Experiments

5.3.1 Datasets and Evaluation Protocols

We conduct experiments on both PASCAL VOC (07 + 12) [37] and MS COCO datasets [90]. For fair comparison, we follow the data split construction and evaluation protocol used in previous works [62]. The PASCAL VOC dataset contains 20 categories. We use the same 3 base/novel splits with TFA [162] and refer them as Novel Split 1,2, 3. Each split contains 15 base classes and 5 novel classes. Each novel class has K annotated instances, where $K = 1, 2, 3, 5, 10$. We report AP50 of the novel categories (nAP50) on VOC07 test set. For MS COCO, the 60 categories disjoint with PASCAL VOC are used as base classes while the remaining 20 classes are used as novel classes. We evaluate our method on shot 1,2,3,5,10,30 and COCO-style AP of the novel classes is adopted as the evaluation metrics.

5.3.2 Implementation Details

Feature generation methods like ours in theory can be built on top of many few-shot object detectors. In our experiments, we use the pre-trained

Faster-RCNN [132] with ResNet-101 [56] following previous work DeFRCN [122]. The dimension of the extracted RoI feature is 2048. For our feature generation model, the encoder consists of three fully-connected (FC) layers and the decoder consists of two FC layers, both with 4096 hidden units. LeakyReLU and ReLU 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. Our semantic embeddings are extracted from a pre-trained CLIP [124] model in all main experiments. An additional experiment using Word2Vec [106] embeddings is reported in Section 5.4.2. After the VAE is trained on the base set with various augmented object boxes , we use the trained decoder to generate $k = 30$ features per class and incorporate them into the fine-tuning stage of the DeFRCN model. We set the function $g(\cdot)$ in Equation 5.2 to a simple linear function $g(x) = w * x + b$ which maps an input IoU score x to the norm of the new latent code. Note that x is in range $[0.5, 1]$ and the norm of the latent code of our VAE before the rescaling typically centers around $\sqrt{512}$ (512 is the dimension of the latent code). We empirically choose $g(\cdot)$ such that the new norm ranges from $\sqrt{512}$ to $5 * \sqrt{512}$. For each feature generation iteration, we gradually increase the value of the controlling parameter β in Equation 5.4 with an interval of 0.75.

5.3.3 Few-shot Detection Results

We use the generated features from our VAE model together with the few-shot samples to fine-tune DeFRCN. We report the performance of two models: "Vanilla-VAE" denotes the performance of the model trained with generated features from a vanilla VAE trained on the base set of ground-truth bounding boxes and "Norm-VAE" denotes the performance of the model trained with features generated from our proposed Norm-VAE model.

PASCAL VOC Table 5.1 shows our results for all three random novel splits from PASCAL VOC. Simply using a VAE model trained with the original data outperforms the state-of-the-art method DeFRCN in all shot and split on PASCAL VOC benchmark. In particular, vanilla-VAE improves DeFRCN by 3.7% for 1-shot and 4.3% for 3-shot on Novel Split 1. Using additional data from our proposed Norm-VAE model consistently improves the results across all settings. We provide qualitative examples in the supplementary material.

	Novel Split 1					Novel Split 2					Novel Split 3				
Method	1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
TFA w/ fc [162]	36.8	29.1	43.6	55.7	57.0	18.2	29.0	33.4	35.5	39.0	27.7	33.6	42.5	48.7	50.2
TFA w/ cos [162]	39.8	36.1	44.7	55.7	56.0	23.5	26.9	34.1	35.1	39.1	30.8	34.8	42.8	49.5	49.8
MPSR [172]	41.7	-	51.4	55.2	61.8	24.4	-	39.2	35.1	39.9	47.8	-	42.3	48.0	49.7
FsDetView [177]	24.2	35.3	42.2	49.1	57.4	21.6	24.6	31.9	37.0	45.7	21.2	30.0	37.2	43.8	49.6
FSCE [152]	44.2	43.8	51.4	61.9	63.4	27.3	29.5	43.5	44.2	50.2	37.2	41.9	47.5	54.6	58.5
CME [84]	41.5	47.5	50.4	58.2	60.9	27.2	30.2	41.4	42.5	46.8	34.3	39.6	45.1	48.3	51.5
SRR-FSD [200]	47.8	50.5	51.3	55.2	56.8	32.5	35.3	39.1	40.8	43.8	40.1	41.5	44.3	46.9	46.4
Halluc. [197]	45.1	44.0	44.7	55.0	55.9	23.2	27.5	35.1	34.9	39.0	30.5	35.1	41.4	49.0	49.3
FSOD-MC [40]	40.1	44.2	51.2	62.0	63.0	33.3	33.1	42.3	46.3	52.3	36.1	43.1	43.5	52.0	56.0
FADI [18]	50.3	54.8	54.2	59.3	63.2	30.6	35.0	40.3	42.8	48.0	45.7	49.7	49.1	48.3	51.5
CoCo-RCNN [99]	43.9	44.5	53.1	64.6	65.5	29.4	31.3	43.8	44.3	51.8	39.1	43.9	47.2	54.7	60.3
MRSN [100]	47.6	48.6	57.8	61.9	62.6	31.2	38.3	46.7	47.1	50.6	35.5	30.9	45.6	54.4	57.4
FCT [51]	49.9	57.1	57.9	63.2	67.1	27.6	34.5	43.7	49.2	51.2	39.5	54.7	52.3	57.0	58.7
Pseudo-Labelling [66]	54.5	53.2	58.8	63.2	65.7	32.8	29.2	50.7	49.8	50.6	48.4	52.7	55.0	59.6	59.6
DeFRCN [122]	56.3	60.3	62.0	67.0	66.1	35.7	45.2	51.5	54.1	53.3	54.5	55.6	56.6	60.8	62.7
Vanila-VAE (Ours)	60.0	63.3	66.3	68.3	67.1	39.3	46.2	52.7	53.5	53.4	56.0	58.8	57.1	62.6	63.6
Norm-VAE (Ours)	**62.1**	**64.9**	**67.8**	**69.2**	**67.5**	**39.9**	**46.8**	**54.4**	**54.2**	**53.6**	**58.2**	**60.3**	**61.0**	**64.0**	**65.5**

Table 5.1: **Few-shot object detection performance (nAP50) on PASCAL VOC dataset**. We evaluate the performance on three different splits. Our method consistently improves upon the baseline for all three splits across all shots. Best performance in bold.

	nAP						nAP75					
Method	1	2	3	5	10	30	1	2	3	5	10	30
TFA w/ fc [162]	2.9	4.3	6.7	8.4	10.0	13.4	2.8	4.1	6.6	8.4	9.2	13.2
TFA w/ cos [162]	3.4	4.6	6.6	8.3	10.0	13.7	3.8	4.8	6.5	8.0	9.3	13.2
MPSR [172]	2.3	3.5	5.2	6.7	9.8	14.1	2.3	3.4	5.1	6.4	9.7	14.2
FADI [18]	5.7	7.0	8.6	10.1	12.2	16.1	6.0	7.0	8.3	9.7	11.9	15.8
FCT [51]	-	7.9	-	-	17.1	21.4	-	7.9	-	-	17.0	22.1
Pseudo-Labelling [66] †	-	-	-	-	17.8	**24.5**	-	-	-	-	**17.8**	**25.0**
DeFRCN [122]	6.6	11.7	13.3	15.6	18.7	22.4	7.0	12.2	13.6	15.1	17.6	22.2
Vanilla-VAE (ours)	8.8	13.0	14.1	**15.9**	**18.7**	22.5	7.9	12.5	13.4	15.1	17.6	22.2
Norm-VAE (ours)	**9.5**	**13.7**	**14.3**	**15.9**	**18.7**	22.5	**8.8**	**13.7**	**14.2**	**15.3**	**17.8**	22.4

Table 5.2: **Few-shot detection performance for the novel classes on MS COCO dataset**. Our approach outperforms baseline methods in most cases, especially in low-shot settings ($K < 10$). † applies mosaic data augmentation introduced in [13] during fine-tuning. Best performance in bold.

MS COCO Table 5.2 shows the FSOD results on MS COCO dataset. Our generated features bring significant improvements in most cases, especially in low-shot settings ($\text{K} \leq 10$). For example, Norm-VAE brings a 2.9% and a 2.0% nAP improvement over DeFRCN in 1-shot and 2-shot settings, respectively. Pseudo-Labeling is better than our method in higher shot settings. However, they apply mosaic data augmentation [13] during fine-tuning.

5.4 Analyses

	Data	1-shot	2-shot	3-shot
DeFRCN [122]	-	56.3	60.3	62.0
VAE	Orginal	60.0	63.3	66.3
VAE	Augmented	60.1	62.7	66.4
Norm-VAE	Augmented	**62.1**	**64.9**	**67.8**

Table 5.3: **Performance comparisons between vanilla VAE and Norm-VAE on PASCAL VOC dataset**. Training a the vanilla VAE with the augmented data does not bring performance improvement. One possible reason is that the generated samples are not guaranteed to be diverse even with sufficient data.

5.4.1 Effectiveness of Norm-VAE

We compare the performance of Norm-VAE with a baseline vanilla VAE model that is trained with the same set of augmented data. As shown in Table 5.3, using the vanilla VAE with more training data does not bring performance improvement compared to the VAE model trained with the base set. This suggests that training with more diverse data does not guarantee diversity in generated samples *w.r.t.* a specific property. Our method, by contrast, improves the baseline model by 1.3% $\sim$ 1.9%, which demonstrates the effectiveness of our proposed Norm-VAE.

5.4.2 Performance Using Different Semantic Embeddings

We use CLIP [124] features in our main experiments. In Table 5.4, we compare this model with another model trained with Word2Vec [106] on PASCAL VOC dataset. Note that CLIP model is trained with 400M pairs

Method	Semantic Embedding	Novel Split 1			Novel Split 2			Novel Split 3		
		1-shot	2-shot	3-shot	1-shot	2-shot	3-shot	1-shot	2-shot	3-shot
DeFRCN [122]	-	56.3	60.3	62.0	35.7	45.2	51.5	54.5	55.6	56.6
Vanilla VAE	Word2Vec	60.4	62.9	**66.7**	38.7	45.2	52.9	55.6	58.7	57.9
Norm-VAE	Word2Vec	**61.6**	**63.4**	66.3	**40.7**	**46.4**	**53.3**	**56.8**	**59.0**	**60.2**
Vanilla VAE	CLIP	60.0	63.3	66.3	39.3	46.2	52.7	56.0	58.8	57.1
Norm-VAE	CLIP	**62.1**	**64.9**	**67.8**	**39.9**	**46.8**	**54.4**	**58.2**	**60.3**	**61.0**

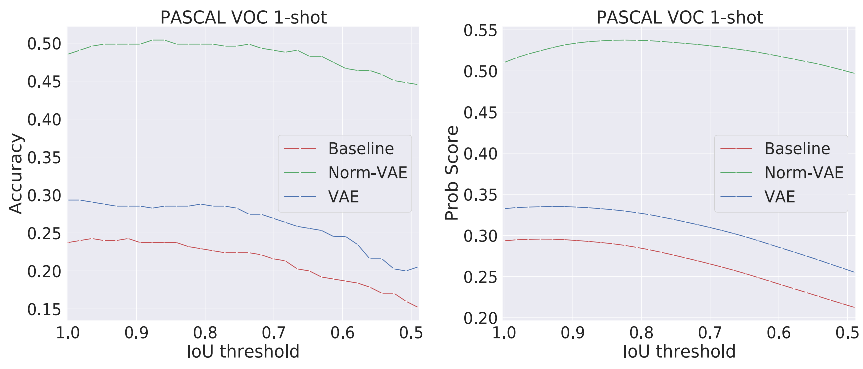
Table 5.4: **FSOD Performance of VAE models trained with different class semantic embeddings**. CLIP [124] is trained with 400M pairs (image and its text title) collected from the web while Word2Vec [106] is trained with only text data.

(image and its text title) collected from the web while Word2Vec is trained with only text data. Our Norm-VAE trained with Word2Vec embedding achieves similar performance to the model trained with CLIP embedding. In both cases, the model outperform the state-of-the-art FSOD method in all settings.

5.4.3 Robustness against Inaccurate Localization

In this section, we conduct experiments to show that our object detector trained with features with diverse crop-related variation is more robust against inaccurate bounding box localization. Specifically, we randomly select 1000 testing instances from PASCAL VOC test set and create 30 augmented boxes for each ground-truth box. Each augmented box is created by enlarging the ground-truth boxes by x% for each dimension where x ranges from 0 to 30. The result is summarized in Figure 5.3 where "Baseline" denotes the performance of DeFRCN[122], "VAE" is the performance of the model trained with features generated from a vanilla VAE, and "Norm-VAE" is the model trained with generated features from our proposed model.

Figure 5.3 (a) shows the classification accuracy of the object detector on the augmented box as the IoU score between the augmented bounding box and the ground-truth box decreases. For both the baseline method DeFRCN and the model trained with features from a vanilla VAE, the accuracy drops by $\sim 10\%$ as the IoU score decreases from 1.0 to 0.5. These results suggest that these models perform much better for boxes that have higher IoU score *w.r.t.* the ground-truth boxes. Our proposed method has



(a) Accuracy (b) Probability score

Figure 5.3: **Classification accuracy and probability score of the object detector on the augmented box**. We compare between the baseline DeFRCN [122], the model trained with features from vanilla VAE and our proposed Norm-VAE. By generating features with diverse crop-related variations, we increase the object detector's robustness against inaccurate object box localization.

higher robustness to these inaccurate boxes: the accuracy of the model trained with features from Norm-VAE only drops by $\sim 5\%$ when IoU score decreases from 1 to 0.5.

Figure 5.3 (b) plots the average probability score of the classifier on the ground-truth category as the IoU score decreases. Similarly, the probability score of both baseline DeFRCN and the model trained with features from a vanilla VAE drops around 0.08 as the IoU score decreases from 1.0 to 0.5. The model trained with features from Norm-VAE, in comparison, has more stable probability score as the IoU threshold decreases.

Method	1-shot	2-shot	3-shot
DeFRCN[122]	16.6	13.3	15.2
Ours (↑ Improvement)	18.8 (↑2.2)	16.4 (↑ 3.1)	19.2 (↑4.0)

Table 5.5: **AP50$\sim$75 of our method and DeFRCN on PASCAL VOC dataset**. AP 50$\sim$75 refers to the average precision computed on the proposals with the IoU thresholds between 50% and 75% and discard the proposals with IoU scores larger than 0.75, i.e., only "*hard*" cases.

5.4.4 Performance on Hard Cases

In Table 5.5, we show AP 50∼75 of our method on PASCAL VOC dataset (Novel Split 1) in comparison with the state-of-the-art method DeFRCN. Here AP 50∼75 refers to the average precision computed on the proposals with the IoU thresholds between 50% and 75% and discard the proposals with IoU scores (*w.r.t.* the ground-truth box) larger than 0.75. Thus, AP 50∼75 implies the performance of the model in *"hard"* cases where the proposals do not significantly overlap the ground-truth object boxes. In this extreme test, the performance of both models are worse than their AP50 counterparts (Table 5.1), showing that FSOD methods are generally not robust to those hard cases. Our method mitigates this issue, outperforming DeFRCN by substantial margins. However, the performance is still far from perfect. Addressing these challenging cases is a fruitful venue for future FSOD work.

5.4.5 Detection Results for Inaccurate Bounding Boxes

In this section, we provide qualitative visualizations of the detected objects of the 1-shot model on PASCAL VOC Novel Split 1. As shown in Figure 5.4, for each input image, the blue box is the original prediction result from the object detector. We then randomly create an augmented bounding box based on the ground-truth bounding box and input the augmented box to the classifier of the object detector. The prediction result on the augmented box is denoted as the yellow box. For the examples shown in the figure, the baseline DeFRCN model [122] and the model trained with features from a vanilla VAE predict the class labels correctly on the original input boxes while both fail on the augmented boxes. By contrast, the model trained with features from Norm-VAE can classify both the original box and the augmented box correctly. As can be seen, crop-related variation is crucial for object detection and our method can enhance the object detector's robustness against the variation successfully.

5.4.6 Number of Generated Samples

In our main experiment, we generate 30 samples per class and use them together with the original few-shot samples to fine-tune the object detector. In this section, we investigate the impact of the number of the gen-

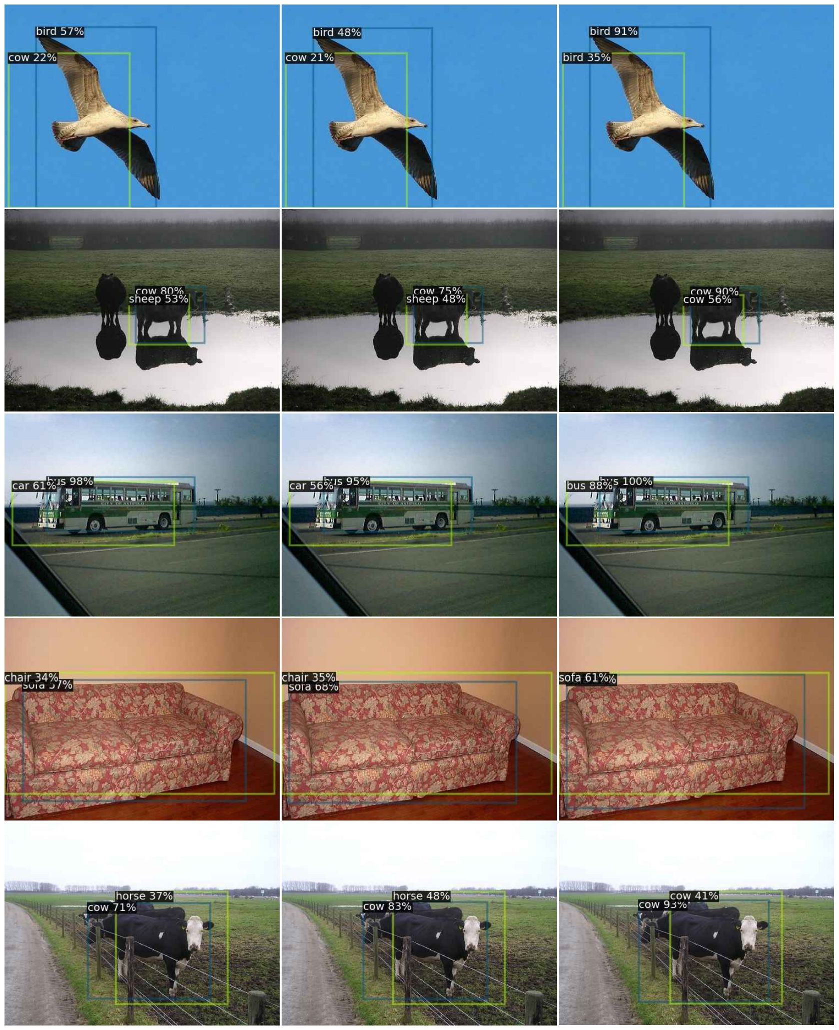


Figure 5.4: **Qualitative visualizations of the detected objects on PASCAL Novel Split 1**. "Vanilla-VAE" denotes the model trained with features generated from a vanilla VAE and "Norm-VAE" denotes the model trained with features generated from Norm-VAE. The blue box is the detector's prediction on the original image and the yellow box is the prediction on the augmented box. Our proposed Norm-VAE can generate features that enhance the model's robustness against crop-related variation.

erated samples. Table 5.6 shows the AP50 on PASCAL VOC Novel Split 1 with different numbers of generated features under 1-shot, 2-shot and 3-shot settings. As the number of generated samples increases, the performance gradually improves and then plateaus and drops slightly (less than 0.5% decrease in performance).

# Generated Features	0	5	10	15	20	25	30	35
1-shot	56.3	60.5	61.6	61.8	62.0	61.9	**62.1**	62.0
2-shot	60.3	62.0	63.7	63.6	63.6	64.1	**64.9**	64.5
3-shot	62.0	65.6	67.0	67.2	67.2	**67.8**	**67.8**	67.3

Table 5.6: **Impact of the number of the generated samples under PASCAL VOC Novel Split 1**. As the number of generated samples increases, the performance gradually improves and then saturates and drops slightly.

5.4.7 Visualization of the Detection Results on PASCAL VOC dataset

We show a few visualization results of DeFRCN [122] and our proposed method in Figure 5.5. As can be seen from the figure, the model trained with additional features performs better than DeFRCN. For instance, in the third row, DeFRCN fails to recognize both the two instances of the *"bird"* class while both Vanilla-VAE and Norm-VAE recognize them. It can be seen that with additional data from Norm-VAE, the FSOD model can recognize objects that are undetected with the model trained with just the original training data. The Norm-VAE model is generally more robust in recognizing objects. It works well even when the objects are cropped (2nd row) or small (two bottom rows).

5.4.8 Mapping Function Analyses

We use a simple pre-defined linear function $g(x) = w \times x + b$ to map from an IoU score x to the new norm of a latent code. Here we only consider proposals with IoU scores ranging from 0.5 to 1. Proposals with lower IoU scores are noisy since they contain mostly background areas. With our VAE architecture and the training data, we observe that the norms of the

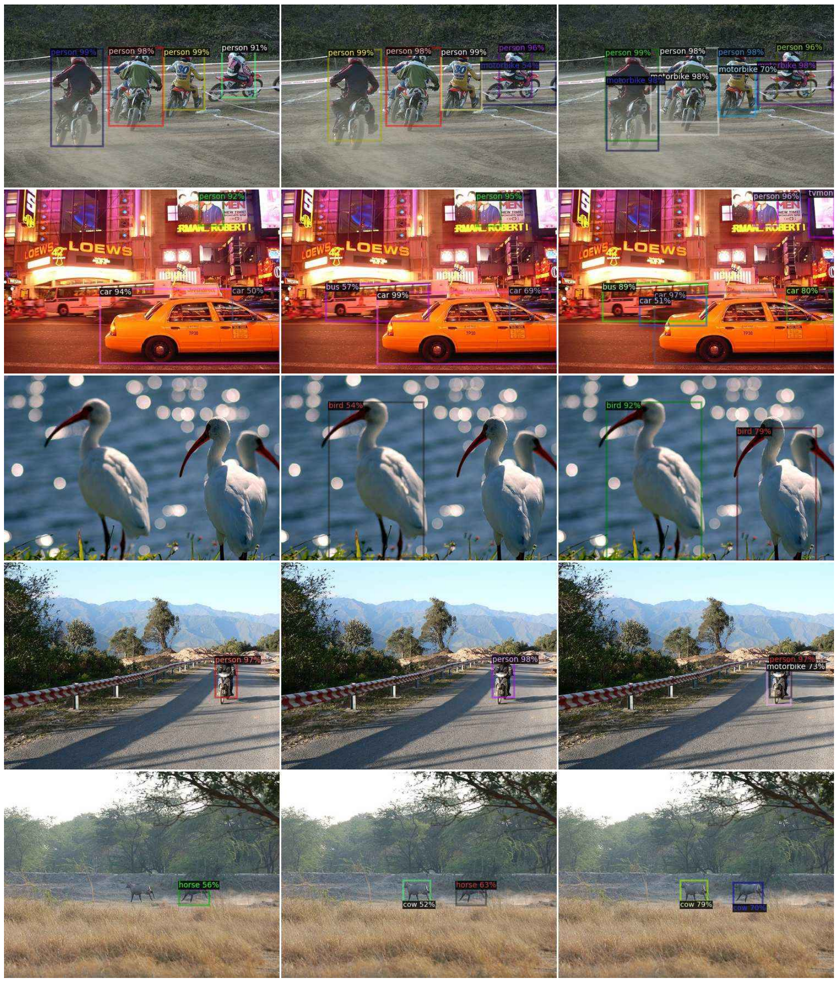


Figure 5.5: **Visualization of the detection results on PASCAL VOC dataset.** The FSOD model trained with additional features performs better than DeFRCN. It works well even when the objects are partially cropped (2nd row) or small (two bottom rows). The detection score threshold is 0.5. Please view in magnification for cases with small objects.

original latent codes are ranged approximately from $\sqrt{512}$ to $5\sqrt{512}$. We would like the rescaled norms to be in the same range and, at the same time, the latent code of an easy proposal has a small norm and the latent code of a hard proposal has a large norm. Thus, we set the parameters of $g(x)$ such that $g(0.5) = 5\sqrt{512}$ and $g(1) = \sqrt{512}$.

We also conduct experiments with different ranges and the results are shown in Table 5.7. Note that here $\sqrt{512}$ is a scaling constant that corresponds to the number of dimensions ($N = 512$) of the latent space. As can be seen from the table, we observe better performance when the IoU score inversely correlates with the latent norm. In this case, a proposal with a low IoU score (i.e., hard case) has a higher latent norm and is placed further away from the origin. A possible reason is that features of hard instances often exhibit higher variance. Thus, it is more optimal to use latent codes with larger norms to represent them [105].

	$g(1)$	$g(0.5)$	AP50
Inverse Correlation	$1\times\sqrt{512}$	$2\times\sqrt{512}$	61.6
	$1\times\sqrt{512}$	$5\times\sqrt{512}$	**62.1**
	$1\times\sqrt{512}$	$10\times\sqrt{512}$	61.8
Correlation	$2\times\sqrt{512}$	$1\times\sqrt{512}$	60.6
	$5\times\sqrt{512}$	$1\times\sqrt{512}$	61.3
	$10\times\sqrt{512}$	$1\times\sqrt{512}$	60.6

Table 5.7: **Performance with different configurations of the mapping function**. We conduct experiments using different coefficients for function $g(\cdot)$, which defines the value range of the new norm of the latent code.

5.4.9 Details on Generating Augmented Training Data

We extract the image features from image crops from the base classes and use them to train a feature generator to generate features for the novel classes. Specifically, we apply the RoI head feature extractor on the ground-truth bounding box b_i from the base classes to get the RoI feature f_i. To enrich the diversity of the RoI feature, we randomly create N additional augmented bounding boxes by randomly moving the starting point and the ending point of the original box, annotated as $\{b_i^1, b_i^2, ...b_i^N\}$. These augmented bounding boxes overlap the ground-truth bounding box differently and have different IoU scores. With a set of augmented bounding boxes $\{b_i^1, b_i^2, ...b_i^N\}$, we extract the corresponding RoI features $\{f_i^1, f_i^2, ...f_i^N\}$ and use them to train our VAE model.

5.5 Conclusion

We tackle the lack of crop-related variability in the training data of FSOD, which makes the model not robust to different object proposals of the same object instance. To this end, we propose a novel VAE model that can generate features with increased crop-related diversity. Experiments show that such increased diversity in the generated samples significantly improves the current state-of-the-art FSOD performance for both PASCAL VOC and MS COCO datasets. Our proposed VAE model is simple, easy to implement, and allows modifying the difficulty levels of the generated samples. In general, generative models whose outputs can be manipulated according to different properties, are crucial to various frameworks and applications. In future work, we plan to address the following limitations of our work: 1) We bias the decoder to increase the diversity in generated samples instead of explicitly enforcing it. 2) Our proposed method is designed to generate visual features of object boxes for FSOD. Generating images might be required in other applications. Another direction to extend our work is to represent other variational factors in the embedding space to effectively diversify generated data.

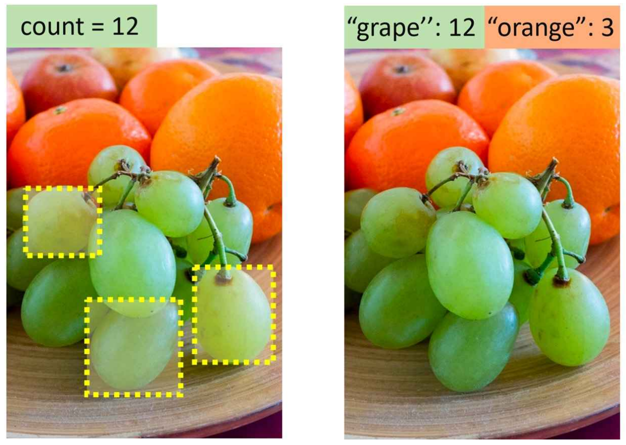
Chapter 6

Generating Class Prototypes for Zero-Shot Object Counting

6.1 Overview

In this Chapter, we present a method for generating reliable data that can serve as object templates for object counting. Object counting aims to infer the number of objects in an image. Most of the existing methods focus on counting objects from specialized categories such as human crowds [138], cars [110], animals [7], and cells [178]. These methods count only a single category at a time. Recently, class-agnostic counting [97, 128, 146] has been proposed to count objects of arbitrary categories. Several human-annotated bounding boxes of objects are required to specify the objects of interest (see Figure 6.1a). However, having humans in the loop is not practical for many real-world applications, such as fully automated wildlife monitoring systems or visual anomaly detection systems.

A more practical setting, exemplar-free class-agnostic counting, has been proposed recently by Ranjan *et al.*[127]. They introduce RepRPN, which first identifies the objects that occur most frequently in the image, and then uses them as exemplars for object counting. Even though RepRPN does not require any annotated boxes at test time, the method simply counts objects from the class with the highest number of instances. Thus, it can not be used for counting a specific class of interest. The method is only suitable for counting images with a single dominant object class, which limits the potential applicability.



(a) Few-shot Counting (b) Zero-Shot Counting

Figure 6.1: Our proposed task of zero-shot object counting (ZSC). Traditional few-shot counting methods require a few exemplars of the object category (a). We propose zero-shot counting where the counter only needs the class name to count the number of object instances. (b). Few-shot counting methods require human annotators at test time while zero-shot counters can be fully automatic.

Our goal is to build an exemplar-free object counter where we can specify what to count. To this end, we introduce a new counting task in which the user only needs to provide the name of the class for counting rather than the exemplars (see Figure 6.1b). In this way, the counting model can not only operate in an automatic manner but also allow the user to define what to count by simply providing the class name. Note that the class to count during test time can be arbitrary. For cases where the test class is completely unseen to the trained model, the counter needs to adapt to the unseen class without any annotated data. Hence, we name this setting zero-shot object counting (ZSC), inspired by previous zero-shot learning approaches [10, 199].

To count without any annotated exemplars, we propose finding a few patches in the input image containing the target object to use them as counting exemplars. There are two challenges: 1) how to localize patches that contain the object of interest based on the provided class name, and 2) how to select *good* exemplars for counting. Ideally, good object exemplars are visually representative for most instances in the image, which can benefit the object counter. In addition, we want to avoid selecting patches that contain irrelevant objects or backgrounds, which likely lead to incorrect object counts. To this end, we propose a two-step method that first localizes the class-relevant patches which contain the objects of interest based on the given class name, and then selects among these patches the optimal exemplars for counting. We use these selected exemplars, together with a pre-trained exemplar-based counting model, to achieve exemplar-free object counting.

The first step of our framework involves constructing a class prototype based on the given class name. Essentially, this requires a mapping between the categorical label and its visual feature. We employ pre-trained large language-vision models to accomplish this task via two approaches.

In the first approach, we learn this mapping between language queries and visual features via a conditional variational autoencoder (VAE). This VAE model is trained to generate visual features of object crops for any arbitrary class, conditioned on its semantic embedding extracted from a pre-trained language-vision model [123]. We take the average of the generated features to compute the class prototype, which can be then used to select class-relevant patches through a simple nearest-neighbour lookup scheme. In essence, our VAE-based approach creates a single prototypical feature for each category that can be applied to any images of this class.

However, relying on a single prototypical feature may not sufficiently capture the diversity within categories exhibiting significant intra-class variance, where objects vary in color (e.g., a green apple versus a red apple), shape (e.g., an SUV versus a regular car), scale, or material composition (a wooden versus a fabric chair). To better address this variability, we propose constructing a class prototype specific to each individual image. To achieve this, we leverage recent advancements in text-to-image generative models, i.e., Stable Diffusion [134], for prototype generation. Unlike traditional VAE-based models, Stable Diffusion enables more realistic and diverse sample generation owing to its extensive training data. This capability offers a robust solution to address the variations present in query objects. Specifically, given a query image, we leverage Stable Diffusion to generate a variety of images containing the objects of interest. Then, we select from these generated images the object crops that most closely resemble the query objects and use them for constructing the class prototype. In this way, a unique prototype is constructed specifically for each testing sample, as opposed to the VAE-based approach's reliance on a single universal prototype for all images. We show that using image-specific prototypes for patch selection generally leads to better counting performance, compared to using a single generic categorical prototype.

After obtaining the class-relevant patches, we further select among them the optimal patches to be used as counting exemplars. Here we observe that the feature maps obtained using *good* exemplars and *bad* exemplars often exhibit distinguishable differences. An example of the feature maps obtained with different exemplars is shown in Figure 6.2. The feature map from a *good* exemplar typically exhibits some repetitive patterns (e.g., the dots on the feature map) that center around the object areas while the patterns from a *bad* exemplar are more irregular and occur randomly across the image. Based on this observation, we train a model to measure the goodness of an input patch based on its corresponding feature maps. Specifically, given an arbitrary patch and a pre-trained exemplar-based object counter, we train this model to predict the counting error of the counter when using the patch as the exemplar. Here the counting error can indicate the goodness of the exemplar. After this error predictor is trained, we use it to select those patches with the smallest predicted errors as the final exemplars for counting.

Experiments on the FSC-147 dataset show that our method outperforms the previous exemplar-free counting method[127] by a large margin. We also provide analyses to show that patches selected by our method can be used in other exemplar-based counting methods to achieve exemplar-free counting. In short, our main contributions can be summarized as follows:

- We introduce the task of zero-shot object counting that counts the number of instances of a specific class in the input image, given only the class name and without relying on any human-annotated exemplars.
- We leverage language-vision models to construct class prototypes via two approaches: VAE-based approach and SD-based approach. We show that in both cases the class prototypes can be used to accurately select patches containing objects of interests for counting.
- We introduce an error prediction model to further select the optimal patches that yield the smallest counting errors.
- We verify the effectiveness of our patch selection method on the FSC-147 dataset, through extensive ablation studies and visualization results.

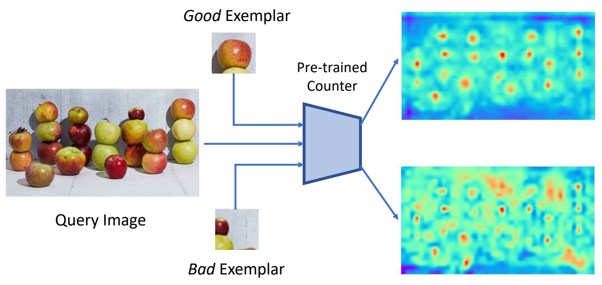


Figure 6.2: Feature maps obtained using different exemplars given a pre-trained exemplar-based counting model. The feature maps obtained using good exemplars typically exhibit some repetitive patterns while the patterns from bad exemplars are more irregular.

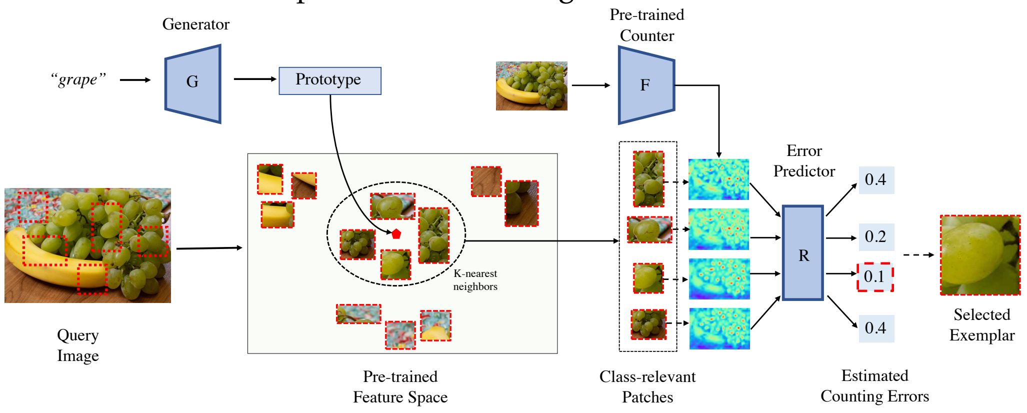


Figure 6.3: Overview of the proposed method. We first obtain a class prototype for the given class name (e.g. grape) in a pre-trained feature space. Then given an input query image, we generate a set of object proposals with a pre-trained RPN and crop the corresponding image patches. We extract the feature embedding for each patch and select the patches whose embeddings are the nearest neighbors of the class prototype as class-relevant patches. Then for each selected class-relevant patch, we use a pre-trained exemplar-based counting model to obtain the intermediate feature maps. Our proposed error predictor then takes the feature maps as input and predicts the counting error (here we use normalized counting errors). We select the patches with the smallest predicted errors as the final exemplar patches and use them for counting.

6.2 Proposed Method

Figure 6.3 summarizes our proposed method. Given an input query image and a class label, we first use a generative model to construct a class prototype for the given class in a pre-trained feature space. We then randomly sample a number of patches of various sizes and extract the feature embedding for each patch. The class-relevant patches are those patches whose embeddings are the nearest neighbors of the class prototype in the embedding space. We further use an error predictor to select the patches with the smallest predicted errors as the final exemplars for counting. We use the selected exemplars in an exemplar-based object counter to infer the object counts. For the rest of the chapter, we denote this exemplar-based counter as the “base counting model". We will first describe how we train this base counting model and then present the details of our patch selection method.

6.2.1 Training Base Counting Model

We train our base counting model using abundant training images with annotations. Similar to previous works [128, 146], the base counting model uses the input image and the exemplars to obtain a density map for object counting. The model consists of a feature extractor F and a counter C. Given a query image I and an exemplar B of an arbitrary class c, we input I and B to the feature extractor to obtain the corresponding output, denoted as $F(I)$ and $F(B)$ respectively. $F(I)$ is a feature map of size $d * h_I * w_I$ and $F(B)$ is a feature map of size $d * h_B * w_B$. We further perform global average pooling on $F(B)$ to form a feature vector b of d dimensions.

After feature extraction, we obtain the similarity map S by correlating the exemplar feature vector b with the image feature map $F(I)$. Specifically, if $w_{ij} = F_{ij}(I)$ is the channel feature at spatial position (i, j), S can be computed by:

$$S_{ij}(I, B) = w_{ij}^T b. \tag{6.1}$$

In the case where n exemplars are given, we use Eq. 6.1 to calculate n similarity maps, and the final similarity map is the average of these n similarity maps.

We then concatenate the image feature map $F(I)$ with the similarity map S, and input them into the counter C to predict a density map D.

The final predicted count N is obtained by summing over the predicted density map D:

$$N = \sum_{i,j} D_{(i,j)}, \tag{6.2}$$

where $D_{(i,j)}$ denotes the density value for pixel (i,j). The supervision signal for training the counting model is the L_2 loss between the predicted density map and the ground truth density map:

$$L_{\text{count}} = \|D(I,B) - D^*(I)\|_2^2, \tag{6.3}$$

where D^* denotes the ground truth density map.

6.2.2 Zero-shot Object Counting

In this section, we describe how we count objects of any unseen category given only the class name without access to any exemplar. Our strategy is to select a few patches in the image that can be used as exemplars for the base counting model. These patches are selected such that: 1) they contain the objects that we are counting and 2) they benefit the counting model, i.e., lead to small counting errors.

Selecting Class-relevant Patches

To select patches that contain the objects of interest, we first generate a class prototype based on the given class name. The class prototype can be considered as a class center representing the patch-level feature distribution of the corresponding class in an embedding space. Then we use the generated class prototype to select the class-relevant patches from a set of object patches cropped from the testing image.

Specifically, we introduce two ways of generating prototypes, i.e., generating semantics prototypes using conditional VAE and generating visual prototypes using samples from a latent text-to-image diffusion model, i.e., Stable Diffusion.

VAE-based prototype generation. To generate class prototypes, we train a conditional VAE model to generate patch-level visual features for an arbitrary class based on the semantic embedding of the class. This strategy is inspired by previous zero-shot learning approaches [174, 175]. The semantic embedding is obtained from a pre-trained text-vision model

[123] given the corresponding class name. Specifically, we train a VAE model to reconstruct deep features extracted from a pre-trained ImageNet model. The VAE is composed of an Encoder E, which maps a visual feature x to a latent code z, and a decoder G which reconstructs x from z. Both E and G are conditioned on the semantic embedding a .The loss function for training this VAE for an input feature x can be defined as:

$$\begin{aligned} L_V(x) = \mathrm{KL}\left(q(z|x,a) \| p(z|a)\right) \\ -\mathrm{E}_{q(z|x,a)}[\log p(x|z,a)]. \end{aligned} \tag{6.4}$$

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.

We can use the trained VAE to generate the semantics prototype for an arbitrary target class for counting. Specifically, given the target class name y, we first generate a set of features by inputting the respective semantic vector a^y and a noise vector z to the decoder G:

$$\mathbb{G}^y = \{\hat{x} | \hat{x} = G(z, y), z \sim \mathcal{N}(0, I)\}. \tag{6.5}$$

The class prototype p^y is computed by taking the mean of all the features generated by VAE:

$$\mathrm{p}^y = \frac{1}{|\mathbb{G}^y|} \sum\nolimits_{\hat{x} \in \mathbb{G}^y} \hat{x} \tag{6.6}$$

SD-based prototype generation. In addition to VAE-based approach for prototype generation, we further leverage the recent advancements in text-to-image models, i.e., Stable Diffusion, to construct class prototypes from SD-generated images. Compared to classic VAE-based models, Stable Diffusion enables more realistic and diverse sample generation, which allows handling the intra-class variation among query objects more effectively.

Specifically, given the target class name for counting, we first use a pre-trained Stable Diffusion model to generate a set of images with the class name as prompt. We observe that the SD-generated images often contain multiple object instances in various contexts and backgrounds. However, our goal is to obtain a few representative object crops of the target class

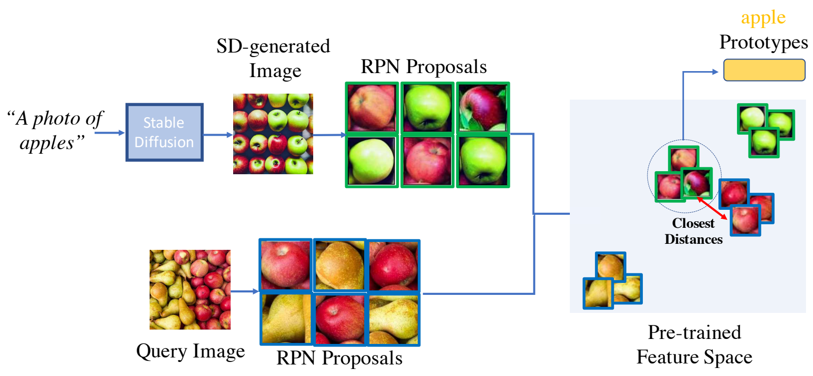


Figure 6.4: Prototype generation using Stable Diffusion.

that can be used to construct reference prototypes. In particular, given a query image, we aim to find a few diffusion-generated object crops that most resemble the target objects in the query image. To do so, we first apply a pre-trained RPN to predict object proposals on both the diffusion-generated images and the query image. Then we compute the pairwise distance between the diffusion-generated object embeddings and the query image's object embeddings. We select the top-k diffusion-generated object embeddings with the nearest mean distance over all query embeddings. We average these k embeddings to construct the visual class prototype. An example is shown in Figure 6.4 where the target objects are red apples. We first obtain a set of object patches containing various crops of apples and select from them a set of red apples to construct the prototype.

Class-relevant patch selection. Using the class prototype, either generated using VAE or Stable Diffusion, we can select the class-relevant patches across the query image. Specifically, we first use a pre-trained RPN to predict object proposals across the query image and extract their corresponding ImageNet features $\{f_1, f_2, ..., f_m\}$. To select the class-relevant patches, we calculate the L_2 distance between the class prototype and the patch embedding, namely $d_i = \|f_i - \mathrm{p}^y\|_2$. Then we select the patches whose embeddings are the nearest neighbors of the class prototype as the class-relevant patches. Since the ImageNet feature space is highly discriminative, i.e., features close to each other typically belong to the same class, the selected patches are likely to contain the objects of the target class.

Selecting Exemplars for Counting

Given a set of class-relevant patches and a pre-trained exemplar-based object counter, we aim to select a few exemplars from these patches that are optimal for counting. To do so, we introduce an error prediction network that predicts the counting error of an arbitrary patch when the patch is used as the exemplar. The counting error is calculated from the pre-trained counting model. Specifically, to train this error predictor, given a query image $\bar{I}$ and an arbitrary patch $\bar{B}$ cropped from $\bar{I}$, we first use the base counting model to get the image feature map $F(\bar{I})$, similarity map $\bar{S}$, and the final predicted density map $\bar{D}$. The counting error of the base counting model can be written as:

$$\epsilon = |\sum_{i,j} \bar{D}_{(i,j)} - \bar{N}^{*}|, \tag{6.7}$$

where $\bar{N}^{*}$ denotes the ground truth object count in image $\bar{I}$. ϵ can be used to measure the goodness of $\bar{B}$ as an exemplar for $\bar{I}$, i.e., a small ϵ indicates that $\bar{B}$ is a suitable exemplar for counting and vice versa.

The error predictor R is trained to regress the counting error produced by the base counting model. The input of R is the channel-wise concatenation of the image feature map $F(\bar{I})$ and the similarity map $\bar{S}$. The training objective is the minimization of the mean squared error between the output of the predictor $R(F(\bar{I}), \bar{S})$ and the actual counting error produced by the base counting model ϵ.

After the error predictor is trained, we can use it to select the optimal patches for counting. The candidates for selection here are the class-relevant patches selected by the class prototype in the previous step. For each candidate patch, we use the trained error predictor to infer the counting error when it is being used as the exemplar. The final selected patches for counting are the patches that yield the top-s smallest counting errors.

Using the Selected Patches as Exemplars

Using the error predictor, we predict the error for each candidate patch and select the patches that lead to the smallest counting errors. The selected patches can then be used as exemplars for the base counting model to get the density map and the final count. We also conduct experiments to show that these selected patches can serve as exemplars for other exemplar-based counting models to achieve exemplar-free class-agnostic counting.

6.3 Experiments

6.3.1 Implementation Details

Network architecture For the *base counting model*, we use ResNet-50 as the backbone of the feature extractor, initialized with the weights of a pre-trained ImageNet model. The backbone outputs feature maps of 1024 channels. For each query image, the number of channels is reduced to 256 using an 1×1 convolution. For each exemplar, the feature maps are first processed with global average pooling and then linearly mapped to obtain a 256-d feature vector. The counter consists of 5 convolutional and bilinear upsampling layers to regress a density map of the same size as the query image. For the *feature generation model*, both the encoder and the decoder are two-layer fully-connected (FC) networks with 4096 hidden units. LeakyReLU and ReLU are the non-linear activation functions in the hidden and output layers, respectively. The dimensions of the latent space and the semantic embeddings are both set to be 512. For the *error predictor*, 5 convolutional and bilinear upsampling layers are followed by a linear layer to output the counting error.

Dataset We use the FSC-147 dataset [128] to train the base counting model and the error predictor. FSC-147 is the first large-scale dataset for class-agnostic counting. It includes 6135 images from 147 categories varying from animals, kitchen utensils, to vehicles. The categories in the training, validation, and test sets do not overlap. The feature generator is trained on the MS-COCO detection dataset. Note that the previous exemplar-free method [127] also uses MS-COCO to pre-train their counter.

Training details Both the base counting model and the error predictor are trained using the AdamW optimizer with a fixed learning rate of 10^{-5}. The base counting model is trained for 300 epochs with a batch size of 8. We resize the input query image to a fixed height of 384, and the width is adjusted accordingly to preserve the aspect ratio of the original image. Exemplars are resized to 128×128 before being input into the feature extractor. The feature generation model is trained using the Adam

Exemplars	Method	Val Set				Test Set			
		MAE	RMSE	NAE	SRE	MAE	RMSE	NAE	SRE
Exemplar-based	FamNet+ [128]	23.75	69.07	0.52	4.25	22.08	99.54	0.44	6.45
	BMNet [146]	19.06	67.95	0.26	4.39	16.71	103.31	0.26	3.32
	BMNet+ [146]	15.74	58.53	0.27	6.57	14.62	91.83	0.25	2.74
	CounTR [93]	13.13	49.83	0.24	0.45	11.95	91.23	0.23	1.72
	SAFECount [189]	14.46	51.88	0.26	0.91	13.58	91.31	0.25	1.66
	Ours	18.55	61.12	0.30	3.18	20.68	109.14	0.36	7.63
Reference-less	RepRPN [127]	30.40	98.73	-	-	27.45	129.69	-	-
	CounTR [93]	17.40	70.33	0.34	1.64	14.12	108.01	0.29	1.93
	FamNet+ [128] + RPN	42.85	121.59	0.75	6.94	42.70	146.08	0.74	7.14
	BMNet [146] + RPN	37.26	108.54	0.42	5.43	37.22	143.13	0.41	5.31
	BMNet+ [146] + RPN	35.15	106.07	0.41	5.28	34.52	132.64	0.39	5.26
	SAFECount [189] + RPN	34.98	107.46	0.38	5.22	33.89	139.92	0.39	5.34
	Ours + RPN	32.19	99.21	0.38	4.80	29.25	130.65	0.35	4.35
Zero-Shot	Patch-Sel (VAE)	27.47	90.85	0.37	4.52	23.14	114.40	0.34	3.95
	Patch-Sel (SD)	**26.30**	**88.80**	**0.34**	**4.27**	**21.53**	**113.28**	**0.31**	**3.61**

Table 6.1: Quantitative comparisons on the FSC-147 dataset. "RPN" denotes using the top-3 RPN proposals with the highest objectness scores as exemplars. "Patch-Sel (VAE)" and "Patch-Sel (SD)" denotes using patch selection method with VAE-generated prototypes and SD-generated prototypes respectively.

optimizer and the learning rate is set to be 10^{-4}. The semantic embeddings are extracted from CLIP [123]. To select the class-relevant patches, we randomly sample 450 boxes of various sizes across the input query image and select 10 patches whose embeddings are the 10-nearest neighbors of the class prototype. The final selected patches are those that yield the top-3 smallest counting errors predicted by the error predictor.

6.3.2 Evaluation Metrics

We use Mean Average Error (MAE) and Root Mean Squared Error (RMSE) to measure the performance of different object counters. Besides, we follow [113] to report the Normalized Relative Error (NAE) and Squared Relative Error (SRE). In particular, MAE $= \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$; RMSE $= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$; NAE $= \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i-\hat{y}_i|}{y_i}$; SRE $= \sqrt{\frac{1}{n}\sum_{i=1}^{n}\frac{(y_i-\hat{y}_i)^2}{y_i}}$ where n is the number of test images, and y_i and $\hat{y}_i$ are the ground truth and the predicted number of objects for image i respectively.

6.3.3 Comparing Methods

We compare our method with the previous works on class-agnostic counting, which can be categorized into exemplar-based counting methods and reference-less counting methods. Exemplar-based methods include FamNet (Few-shot adaptation and matching Network [128]), BMNet (Bilinear Matching Network [146]), CounTR (Counting TRansformer [93]) and SAFECount (Similarity-Aware Feature Enhancement block for object Counting [189]). These methods require a few human-annotated exemplars as inputs. Reference-less methods, i.e., RepRPN [127] and CounTR [93], do not require annotated boxes at test time. Nevertheless, the class of interest can not be specified, which makes them only suitable for counting images with a single dominant object class. Our proposed zero-shot counting, is a new setup which allows the user to specify what to count by simply providing the class name without any exemplar. We also make exemplar-based methods work in the exemplar-free manner by replacing the human-provided exemplars with the exemplars generated by a pre-trained object detector. Specifically, we use the RPN of Faster RCNN pre-trained on MS-COCO dataset and select the top-3 proposals with the highest objectness score as the exemplars.

6.3.4 Results

Quantitative results. As shown in Table 6.1, the performance of all exemplar-based counting methods drops significantly when replacing human-annotated exemplars with RPN generated proposals. BMNet+ [146], for example, shows an 19.90 error increase *w.r.t.* the test MAE and a 40.81 increase *w.r.t.* the test RMSE. In comparison, the performance gap is much smaller when using our selected patches as exemplars. Our patch selection method with VAE-generated prototype obtains 27.47 MAE on the validation set and 23.14 MAE on the test set. By using the SD-generated class prototype, the error rates can be further reduced, achieving 26.30 MAE on the validation set and 21.53 MAE on the test set. Noticeably, compared with the human-annotated exemplars, the NAE and the SRE on the test set are even reduced when using our selected patches.

Qualitative analysis. In Figure 6.5, we present a few input images, the image patches selected by our method, and the corresponding density maps. Our method effectively identifies the patches that are suitable for

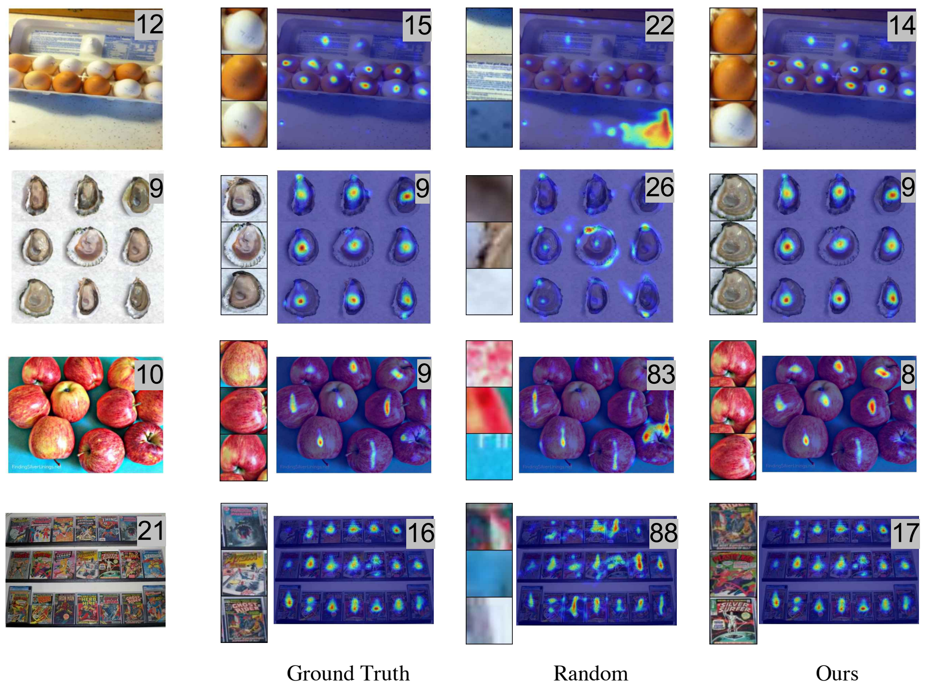


Figure 6.5: Qualitative results on the FSC-147 dataset. We show the counting exemplars and the corresponding density maps of ground truth boxes, randomly selected patches, and our selected patches respectively. Predicted counting results are shown at the top-right corner. Our method accurately identifies suitable patches for counting and the predicted density maps are close to the ground truth density maps.

object counting. The density maps produced by our selected patches are meaningful and close to the density maps produced by human-annotated patches. The counting model with random image patches as exemplars, in comparison, fails to output meaningful density maps and infers incorrect object counts.

In Figure 6.6, we visualize some images from the FSC-147 dataset and the corresponding patches selected by RPN and our method respectively. The RPN-selected patches are the top-3 proposals with the highest objectness scores. As can be seen from the figure, the patches selected by RPN may contain objects not relevant to the provided class name or contain multiple object instances. These patches are not suitable to be used as counting exemplars and will lead to inaccurate counting results. This suggests that choosing counting exemplars based on objectness score is not reliable. In comparison, our proposed method can accurately localize image patches according to the given class name. These selected patches can then be used as counting exemplars and yield meaningul density maps and reasonable counting results.

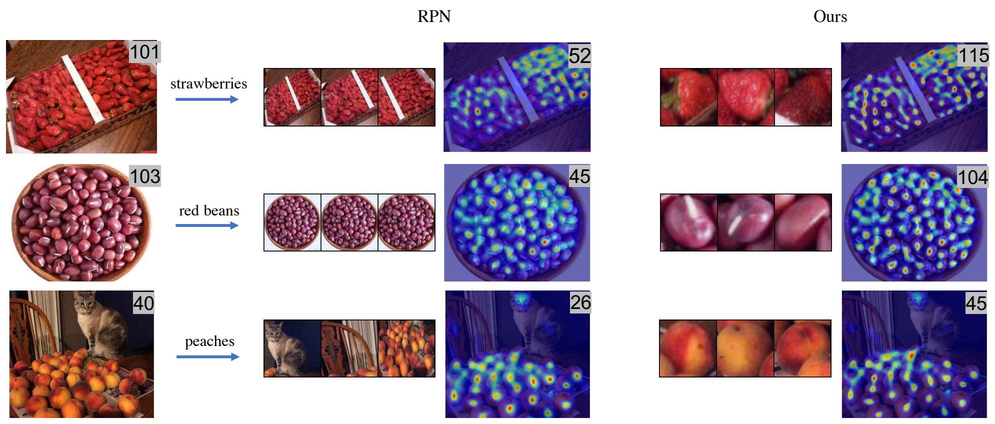


Figure 6.6: Qualitative comparison with top-3 exemplars from RPN. Our proposed method can select patches suitable for counting while RPN-selected patches contain non-relevant objects or multiple object instances.

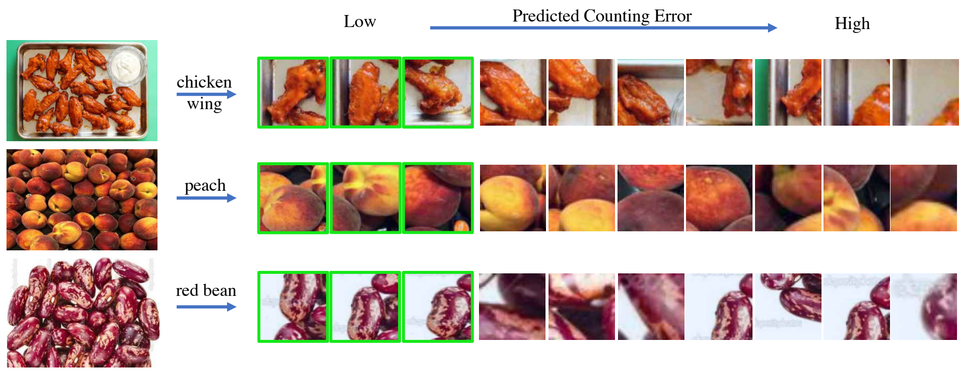


Figure 6.7: Qualitative ablation analysis. All the 10 selected class-relevant patches exhibit some class-specific attributes. They are ranked by the predicted counting errors and the final selected patches with the smallest errors are framed in green.

6.4 Analyses

6.4.1 Ablation Studies

Our proposed patch selection method consists of two steps: the selection of class-relevant patches via a generated class prototype and the selection of the optimal patches via an error predictor. We analyze the contribution of each step quantitatively and qualitatively. Quantitative results are in Table 6.2. We first evaluate the performance of our baseline, i.e. using 3 randomly sampled patches as exemplars without any selection step. As shown in Table 6.2, using the class prototype to select class-relevant patches reduces the error rate by 7.19 and 6.07 on the validation and test set of MAE, respectively. Applying the error predictor can improve the baseline performance by 7.22 on the validation MAE and 7.57 on the test MAE. Finally, applying the two components together further boosts performance, achieving 26.93 on the validation MAE and 22.09 on the test MAE.

We provide further qualitative analysis by visualizing the selected patches. As shown in Figure 6.7, for each input query image, we show 10 class-relevant patches selected using our generated prototype, ranked by their predicted counting error (from low to high). All the 10 selected class-relevant patches exhibit some class specific features. However, not all these patches are suitable to be used as counting exemplars, i.e., some patches only contain parts of the object, and some patches contain some background. By further applying our proposed error predictor, we can identify the most suitable patches with the smallest predicted counting errors.

6.4.2 Generalization to Exemplar-based Methods

Our proposed method can be considered as a general patch selection method that is applicable to other visual counters to achieve exemplar-free counting. To verify that, we use our selected patches as the exemplars for three other different exemplar-based methods: FamNet [128], BMNet and BMNet+ [146]. Figure 6.8 (a) shows the results on the FSC-147 validation set. The baseline uses three randomly sampled patches as the exemplars for the pre-trained exemplar-based counter. By using the generated class prototype to select class-relevant patches, the error rate is reduced

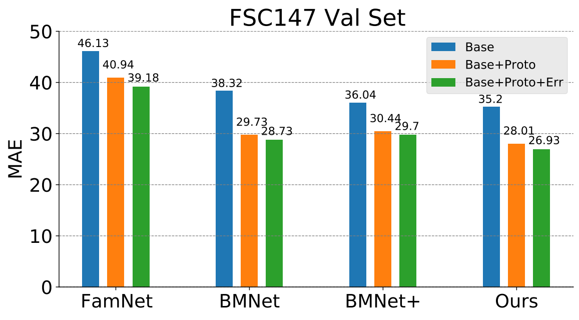
Prototype	Predictor	Val Set				Test Set			
		MAE	RMSE	NAE	SRE	MAE	RMSE	NAE	SRE
-	-	35.20	106.70	0.61	6.68	31.37	134.98	0.52	5.92
✓	-	28.01	88.29	0.39	4.66	25.30	**113.82**	0.40	4.88
-	✓	27.98	**88.62**	0.43	4.59	23.80	128.36	0.40	4.43
✓	✓	**26.93**	88.63	**0.36**	**4.26**	**22.09**	115.17	**0.34**	**3.74**

Table 6.2: Ablation study on each component's contribution to the final results. We show the effectiveness of the two steps of our framework: selecting class-relevant patches via a generated class prototype and selecting optimal patches via an error predictor.

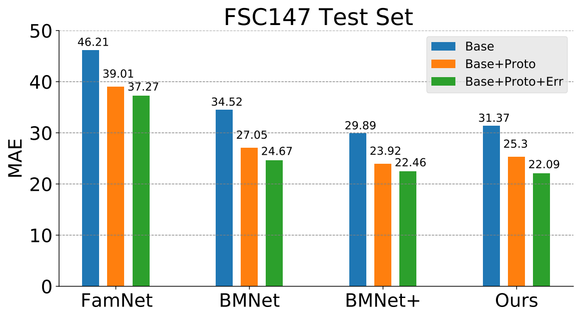
by 5.18, 8.59 and 5.60 on FamNet, BMNet and BMNet+, respectively. In addition, as the error predictor is additionally adopted, the error rate is further reduced by 1.76, 1.00 and 1.08 on FamNet, BMNet and BMNet+, respectively. Similarly, Figure 6.8 (b) shows the results on the FSC-147 test set. Our method achieves consistent performance improvements for all three methods.

6.4.3 Multi-class Object Counting

Our method can count instances of a specific class given the class name, which is particularly useful when there are multiple classes in the same image. In this section, we show some visualization results in this multi-class scenario. As seen in Figure 6.9, our method selects patches according to the given class name and counts instances from that specific class in the input image. Correspondingly, the heatmap highlights the image regions that are most relevant to the specified class. Here the heatmaps are obtained by correlating the exemplar feature vector with the image feature map in a pre-trained ImageNet feature space. Note that we mask out the image region where the activation value in the heatmap is below a threshold when counting. We also show the patches selected using another exemplar-free counting method, RepRPN [127]. The class of RepRPN selected patches can not be explicitly specified. It simply selects patches from the class with the highest number of instances in the image according to the repetition score.



(a)

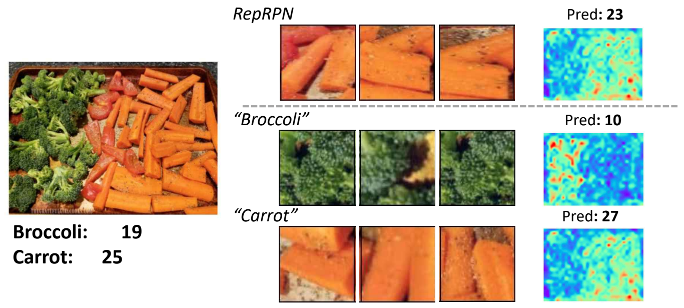


(b)

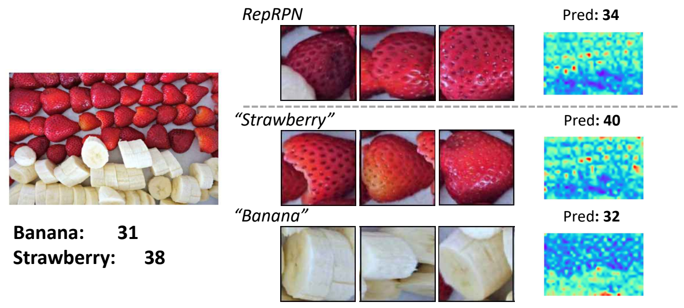
Figure 6.8: Using our selected patches as exemplars for other exemplar-based class-agnostic counting methods (FamNet, BMNet and BMNet+) on FSC-147 dataset. Blue bars are the MAEs of using three randomly sampled patches. Orange bars are the MAEs of using the class prototype to select class-relevant patches as exemplars. Green bars are the MAEs of using the class prototype and error predictor to select optimal patches as exemplars.

6.4.4 Qualitative Comparison between SD-generated Prototypes and VAE-generated Prototypes

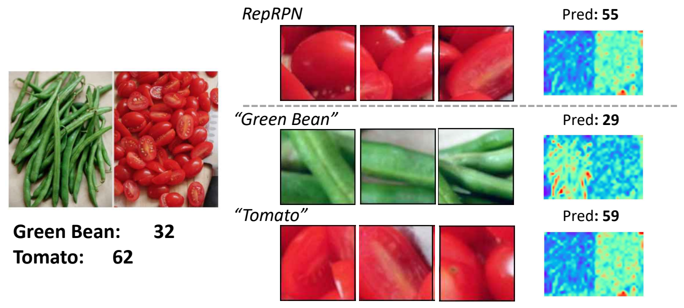
In this section, we provide qualitative comparison between patches selected via SD-generated prototypes and VAE-generated prototypes. As shown in Figure 6.10, we present a few input images and the corresponding patches selected by VAE-generated prototypes and SD-generated prototypes. Although the patches selected by VAE-generated prototypes generally contain the objects of interest, they miss parts of the objects in some cases (e.g., the second patch of *grape*), or contain multiple object instances



(a)



(b)



(c)

Figure 6.9: Visualization results of our method in some multi-class examples. Our method selects patches according to the given class name and the corresponding heatmap highlights the relevant areas.

within one patch (e.g., the second patch of *strawberry*). In comparison, the patches selected by SD-generated prototypes are generally better exemplars for counting, i.e., one patch mostly contains a single complete object instance.

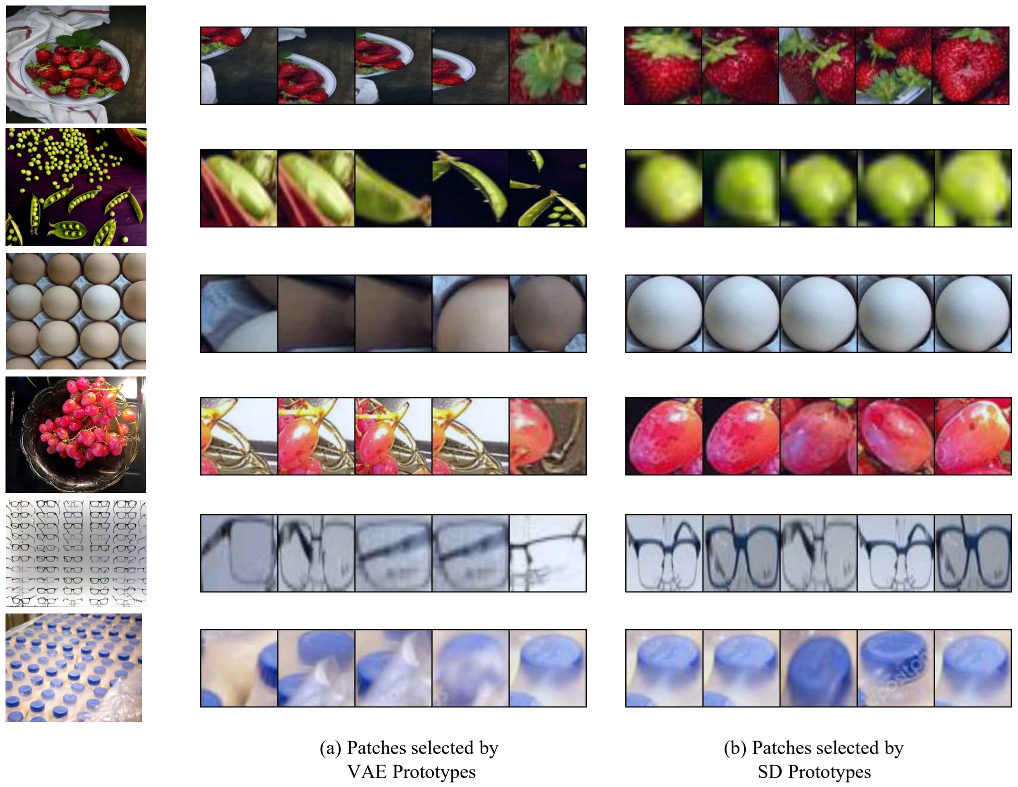
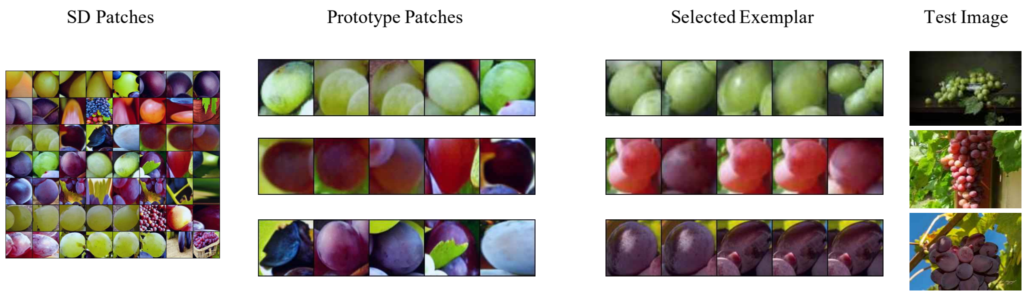


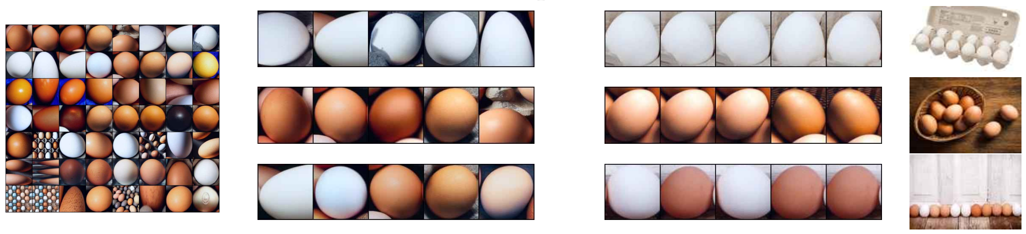
Figure 6.10: Visualization of the patches selected by VAE-generated prototypes and SD-generated prototypes. Patches selected by SD-generated prototypes are of higher quality.

6.4.5 Analysis on SD-generated Prototypes

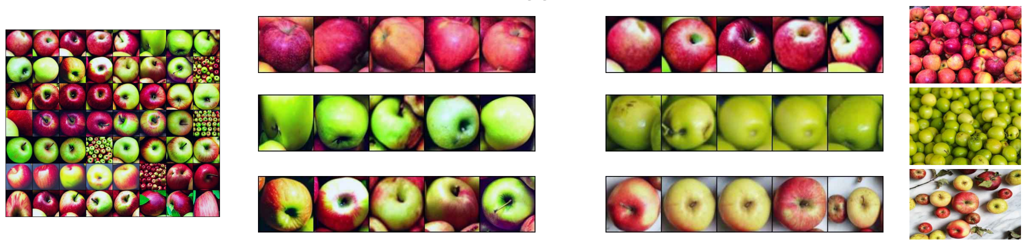
Qualitative Visualization. To generate visual class prototypes, we first use a pre-trained Stable Diffusion to generate a set of object patches for the class of interest. Then given a query image, we select the generated patches that most resemble the target objects in the query image. We compute the average feature embeddings of the selected generated patches to construct the prototype. In Figure 6.11, we demonstrate this process for three different categories, i.e., grapes, eggs, and apples. For each category, we show how we select different patches to construct the prototypes for the given query image. As can be seen in the first column, the set of RPN proposals extracted from SD-generated images exhibit rich vari-



(a) Grapes



(b) Eggs



(c) Apples

Figure 6.11: Visualization of the diffusion-generated patches for prototype generation and the corresponding selected patches from query images. Our method constructs prototypes according to the query images and selects the patches that most resemble the target objects.

ations. Among these proposals, our method selects those that are most relevant to the testing images for constructing prototypes. For example, in the last row of Figure 6.11 (c), only apples with mixed colors are selected to match the objects' colors in the testing image. Compared with VAE-generated prototypes which remain the same for all query images, SD-generated prototypes can better handle the variations of query images and select more accurate counting exemplars.

Number of Patches for Prototype Generation. In our main experiments, we select the top-5 SD-generated patches with the nearest mean distance over query patches, and compute their average features to construct prototypes. In this section, we conduct an ablation study on how the number of patches selected for prototype generation affects the counting performance. Specifically, we select top-5, top-25, top-50 and all patches to construct class prototypes and use them to select exemplars. Results are summarized in Table 6.3. We observe that the performance drops on both the validation set and test set as the number of selected patches increases. The counting errors are highest when using all generated patches to construct prototypes. In this case, the same class prototype is applied for all images of this class, which is not optimal for counting objects with large intra-class diversity. Our method, in comparison, selects the most similar patches based on the query image, which leads to more accurate prototypes.

Prototype	Val Set		Test Set	
Patches	MAE	RMSE	MAE	RMSE
top-5	**27.76**	**97.06**	**21.99**	**113.31**
top-25	28.03	97.87	22.12	113.71
top-50	28.33	98.50	22.34	113.88
all	30.13	101.70	23.37	116.04

Table 6.3: Ablation study on the number of diffusion-generated patches for prototype generation.

6.5 Conclusion

We propose a new task, zero-shot object counting, to count instances of a specific class given only the class name without access to any exemplars.

To address this, we developed a two-step approach that accurately localizes the optimal patches to be used as counting exemplars. We leverage language-vision models to construct class prototypes via two approaches: VAE-based approach and diffusion-based approach. Through these two approaches, we present a comprehensive study of prototype construction at both category and image levels. In the context of our specific task, the diffusion-based image prototype has notably outperformed the category-level prototype constructed via VAE, thanks to its ability to customize prototypes to match object appearances in each image. More generally, our approach of employing language-vision models for zero-shot recognition is applicable in various tasks. In scenarios where data of large-scale generative models do not exist such as medical imaging, or remote sensing, VAE can be a useful choice.

We show that the prototypes can be used to select the patches containing objects of interests. Furthermore, we introduce an error prediction model to select those patches with the smallest predicted errors as the final exemplars for counting. Extensive results demonstrate the effectiveness of our method. We also conduct experiments to show that our selected patches can be incorporated into other exemplar-based counting methods to achieve exemplar-free counting.

Chapter 7

Assessing Sample Quality via the Latent Space of Generative Models

7.1 Overview

In this Chapter, we propose a novel approach to estimate sample quality via the latent space of generative models. Generative models have emerged as powerful modeling tools that can capture diverse and complex distribution from a large training dataset to synthesize new data. A single pre-trained diffusion model[135] can generate thousands of images of "Yorkshire Terrier" or "Notre-Dame de Paris". We aim to answer the question: among the samples generated from the model, how to measure the quality of each individual one? Such an instance-wise quality assessment metric is essential for users and consumers to select samples among the ones provided by those recently released text-to-image models, *e.g.*, DALL-E 2 [126] and Stable Diffusion [135], rather than model-wise metrics such as Frechèt Inception Distance (FID) [57].

For the most part, previous instance-wise evaluation methods [52, 74] rely on a pre-trained feature extractor (*e.g.*, VGG16 [149]) to embed the generated samples and real samples into a common feature space. K-nearest neighbor (k-NN) based approaches are then applied under the assumption that close samples in this feature space correspond to semantically similar images. The realism score [74], for example, measures the maximum of the inverse relative distance of a fake sample in a real k-NN latent sphere. The rarity score [52], on the other hand, measures the minimum radius of a real k-NN sphere that contains the fake latent representation. However, relying on a pre-trained feature extractor suffers from two shortcomings. First, different feature extractors might lead to inconsistent assessment outcomes: the rarity score shows a negative correlation with Frechèt Inception Distance (FID) [57] when using VGG16 as backbone, while the correlation becomes positive under DINO [19] or CLIP [124] backbones. Moreover, these methods are not applicable for domains where a robust, universal feature extractor is not yet available, *e.g.*, 3D shapes, human-drawn art or medical images.

In this work, we propose to assess sample quality from another perspective: instead of using a pre-trained feature space, we directly use *the latent space of the generative models themselves*. The intuition is that the quality of a generated sample directly relates to the amount of the training samples that closely resemble it, and we can infer this information solely by examining the density of the latent space. Specifically, the samples lying in the latent area with dense latent codes are likely to have sufficient training data resembling them while low-density latent areas would correspond to the rare cases in the data manifold. This is because generative models typically map similar data points to similar latent embeddings. The latent embeddings in those low-density areas are less exposed in model training, consequently receiving less supervision, and leading to potentially inferior reconstruction quality.

To this end, we propose a latent density score function to measure the quality of generated samples. Given a pre-trained generative model, our proposed function quantitatively measures the density of a randomly sampled latent code w.r.t. a set of latent codes extracted from the training data. We show that the proposed latent density score highly correlates with the sample quality for various generative models including Variational Autoencoders (VAEs) [72], Generative Adversarial Networks (GANs) [46] and Latent Diffusion Models (LDMs) [135]. Compared with previous quality assessments that require an additional embedding network for feature extraction, our method estimates the sample quality by directly examining the latent space of the generative models, which brings several key advantages: 1) **efficiency**: our method enables quality assessment without generating image pixels, which significantly reduces the computational cost; 2) **generalizability**: our method eliminates the reliance on external feature extractors, which allows for generalization to the domains where a universal pre-trained feature extractor might not exist; 3) **applicability**: our method can be seamlessly incorporated into latent-based image editing and generation methods, which can benefit various downstream tasks.

In short, our main contributions can be summarized as follows:

- We demonstrate that we can directly assess sample quality via the latent space of generative models themselves, while previous quality assessment methods rely on a pre-trained feature extractor to embed real and generated samples to a common space.

- We propose a score function to quantify sample quality by measuring the density in latent space. The proposed function is applicable to various generative models trained on a variety of datasets.

- We show the clear advantages of our proposed method over previous instance-wise evaluation methods, including significantly saving computational cost, generalizing across different domains and facilitating various downstream tasks.

7.2 Latent Density Score

Given a well-trained generative model, *e.g.*, GAN, VAE or latent diffusion model, we aim to estimate the quality of the generated samples by examining the latent space of the model. Let $\mathcal{Z} = \{z_1, z_2, ...z_i\}$ denote a set of latent codes extracted from the training samples, and z_g denote a latent code randomly sampled from the latent space, we measure the latent density of z_g quantitatively by calculating the latent density score as:

$$D(z_g, \mathcal{Z}) = \frac{1}{|\mathcal{Z}|} * \sum_{z_i \in \mathcal{Z}} e^{-\frac{\|z_g - z_i\|^2}{2\sigma^2}}, \tag{7.1}$$

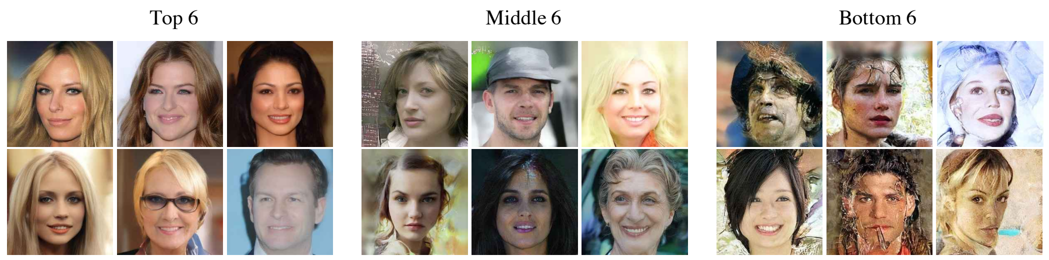
where σ is a hyper-parameter of this score function. Latent density score measures the average Gaussian kernelized Euclidean distance [27] between z_g and each latent code in $\mathcal{Z}$. The score is high when z_g resides in an area where the trained codes are densely distributed. σ controls the relative contribution of each latent code in $\mathcal{Z}$ to the final density value, *i.e.*, using a small σ places more emphasis on the local area surrounding z_g, while applying a large σ places more focus on the global density. In the case where there are multiple local clusters in the latent manifold, different values of σ will lead to different assessment results (see Section 7.5.2).

In GAN-based generative models, truncation trick [16, 63, 71] is a widely used technique to increase the sample fidelity at the cost of lowering the diversity. It works by shifting a randomly sampled code towards the mean latent code. The mean code typically resides in a high-density latent area. In fact, we observe that the proposed latent density score well correlates with the degree of truncation. We analyze this correlation further in Section 7.5.1 and provide more qualitative results in the Supplementary Material. Another highly relevant quality assessment metric is the realism score [74]. The realism score measures the relative distance of a fake sample in a real latent sphere, which is defined by a pre-trained feature extractor. We show that the latent density score behaves similarly with the realism score for images from the domains previously seen by the feature extractor (see Section 7.5.1). However, for images from non-ImageNet-like domains (*e.g.*, medical images and anime-style images) or domains other than 2D images (*e.g.*, 3D shapes), quality assessment with realism score will be infeasible (see Section 7.3.2).

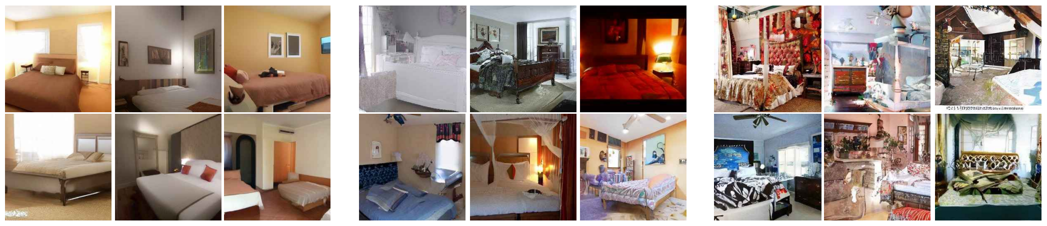
7.3 Experimental Results

7.3.1 Results on Different Generative Models

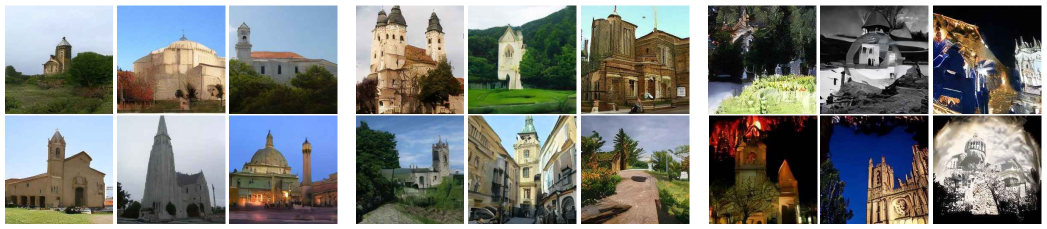
In this section, we provide the experimental results of the proposed metric for various generative models and datasets. We experiment with three types of generative models, *i.e.*, GANs, VAEs and LDMs. For each trained model, we extract latent codes from 60k training samples and calculate the latent density scores for 20k randomly sampled latent codes. In particular, for VAEs and LDMs, we take the output of the image encoder as the latent representation of each real input image. For LDMs, we further flatten the 2D representations (before the denoising process) for computing the latent density score. We use the pre-trained Stable Diffusion v1.5 model [135] as the text-to-image diffusion model. For the unconditional diffusion models, we choose the LDMs pre-trained on CelebA-HQ [26],



(a) Latent Diffusion - CelebA-HQ



(b) Latent Diffusion - LSUN-Bedrooms



(c) Latent Diffusion - LSUN-Churches

Figure 7.1: Top 6, Middle 6 and Bottom 6 generated images in terms of the proposed latent density score on CelebA-HQ, LSUN-Bedrooms and LSUN-Churches for unconditional latent diffusion models. (Zoom-in for best view). The proposed latent density scores highly correlate with the quality of generated images.

LSUN-Bedrooms and LSUN-Churches [190] released by [135]. For GANs, we experiment with StyleGAN2 [65] and StyleGAN2-ADA [64]. To obtain the latent representations, we input vectors sampled from a normal distribution to their mapping networks and extract latent features from the $\mathcal{W}$-space. We use $\sigma = 20$ for computing latent density scores. We analyze the choice of σ and how it affects the results in Section 7.5.2.

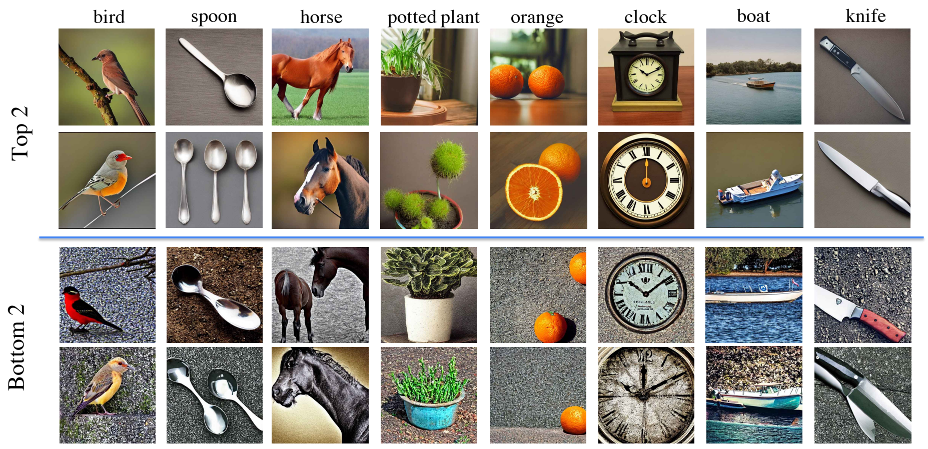
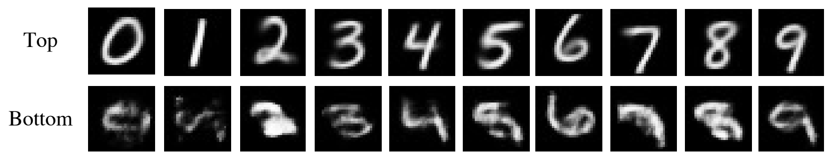


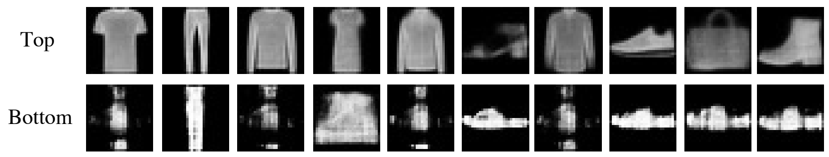
Figure 7.2: Top 2 and bottom 2 Stable Diffusion generated samples for eight classes in terms of the proposed latent density score. Images in the 'top 2' rows are high-resolution images with natural, realistic backgrounds, whereas images in the 'bottom 2' rows contain visual noise and artifacts. The only difference in model configuration for images of top / bottom rows is the initial noise.

Latent Diffusion Models

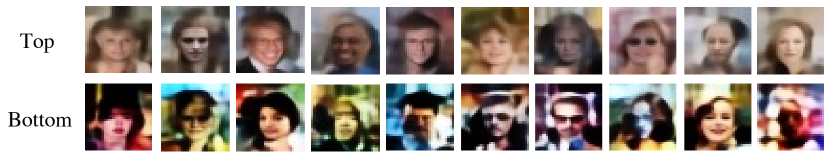
Latent diffusion models [135] use pre-trained autoencoders to construct a low-dimensional latent space, from which the original data can be reconstructed at high fidelity with reduced computational costs. In Figure 7.1, we show images synthesized by unconditional latent diffusion models trained on CelebA-HQ, LSUN-Bedrooms and LSUN-Churches. For each dataset, we show samples using latent codes with the top 6 highest, top 6 lowest and 6 middle latent density scores. As shown in the figure, the proposed latent density scores highly correlate with the quality of generated images. For example, on the CelebA-HQ dataset, we can see human faces generated from codes with high latent density scores are visually realistic with clear hair, eye and eyebrow details, whereas those with low latent density scores are of degraded quality due to blur, artifacts or distorted facial structures. Similarly, on LSUN-Bedrooms and LSUN-Churches, we observe unrealistic artifacts (*i.e.*, distorted textures or inharmonious colors) from images with low latent density scores.



(a) VAE - MNIST



(b) VAE - Fashion-MNIST



(c) VAE - CelebA

Figure 7.3: Top 10 and Bottom 10 generated images in terms of the proposed latent density score on MNIST, Fashion-MNIST and CelebA for VAE. The samples with high latent density scores display clear instances, whereas those with low latent density scores are often distorted / blurred.

Figure 7.2 shows images synthesized by a pre-trained text-to-image diffusion model, *i.e.*, Stable Diffusion, using latent codes with the top 2 highest and lowest latent density scores from eight classes. As shown in the figure, samples with high latent density scores have superior visual quality while latent codes with low scores often lead to erroneous samples. The most obvious failures are the unrealistic backgrounds. For example, the boat images in the 'top 2' rows are high-resolution images with natural, realistic backgrounds, whereas the backgrounds of the boats in the 'bottom 2' rows contain visual noise and artifacts. In some other failure cases, the generated objects exhibit structural integrity artifacts, *i.e.*, the spoons and the clocks. We note that all images here are generated with the same model configuration and the only difference is the initial noise. Previous works [32, 59, 114] have shown that the guidance methods used in the denoising process are essential for improving the quality of generated images. Here we observe that the randomly initialized noise can also affect the sample quality significantly.

VAEs and GANs

Figure 7.3 presents images generated by VAEs using codes with the top 10 highest and top 10 lowest latent density scores. As shown in the figure, our proposed score function is applicable to VAEs as well. For MNIST [81] and Fashion-MNIST [176], for example, we observe the samples with high latent density scores display clear instances from the given class, whereas those with low latent density scores are often distorted / blurred to unrecognizable.

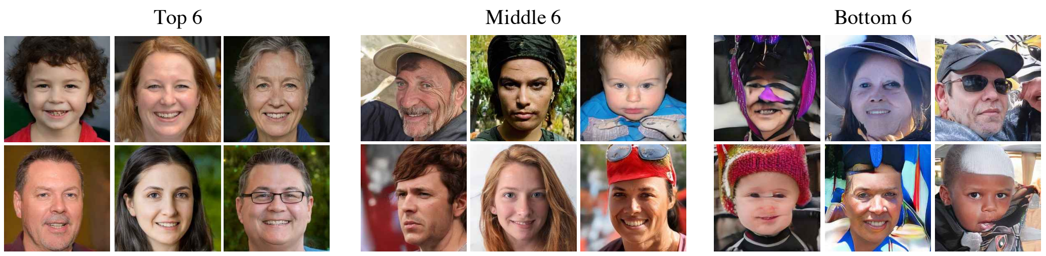
Figure 7.4 shows images generated by StyleGAN2 [65] trained on FFHQ [63], StyleGAN2-ADA [64] trained on AFHQ Dog [26] and StyleGAN2 trained on AFHQ Cat [26] using codes with the top 6 highest, top 6 lowest and 6 middle latent density scores. We observe clear generation quality differences between samples with different scores. For example, on the AFHQ Dog dataset, the samples with high scores show clear, frontal dog faces. On the other hand, the dog faces in samples with low density scores are highly distorted in various ways.

7.3.2 Results on Other Domains and Modalities

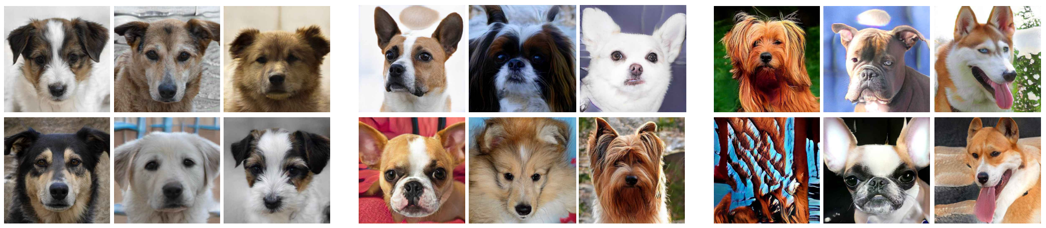
Our proposed metric does not rely on any additional feature extractor, which enables quality assessment in the domains where robust pre-trained models might not be available. In this section, we show the applications of our method on quality assessment for generated 3D shapes and non-ImageNet-like images.

Quality Assessment for 3D Shapes

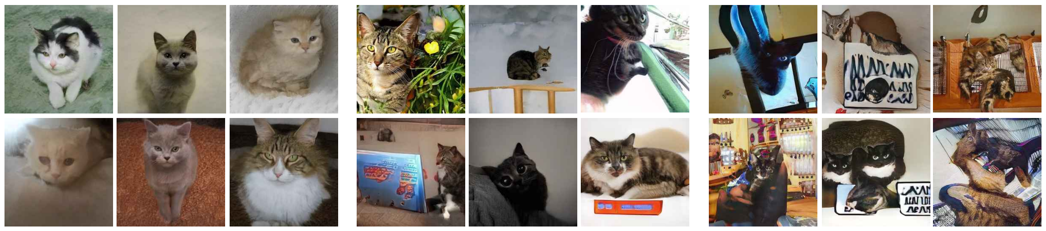
We first show the application of our method on generated 3D shapes. Specifically, we generate shapes for four categories, *i.e.*, airplane, chair, table and rifle, using a StyleGAN2-based 3D shape generation framework, SDF-StyleGAN [198] trained on ShapeNet Core V1 [20]. For each shape category, we extract the latent embeddings in the $\mathcal{W}$ space of SDF-StyleGAN for 30k randomly sampled vectors and compute the corresponding latent density scores. Figure 7.5 visualizes the generated shapes



(a) StyleGAN2 - FFHQ



(b) StyleGAN2-ADA - AFHQ Dog



(c) StyleGAN2 - AFHQ Cat

Figure 7.4: Top 6, Middle 6 and Bottom 6 generated images in terms of the proposed latent density score on FFHQ for StyleGAN2, on AFHQ Dog for StyleGAN2-ADA and on AFHQ Cat for StyleGAN2. (Zoom-in for best view). Samples with high scores are of better quality while samples with low scores are often highly distorted.

with the top 5 highest and lowest scores. We observe that the generated 3D shapes with high scores have better visual quality with plausible 3D shapes and complete geometry structures. The generated 3D shapes with low scores, in contrast, exhibit unrealistic shapes and severe geometry distortion.

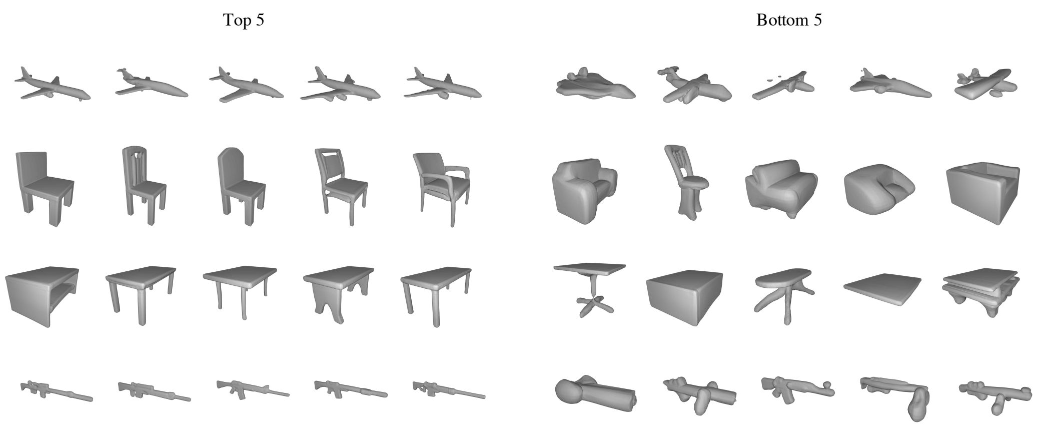
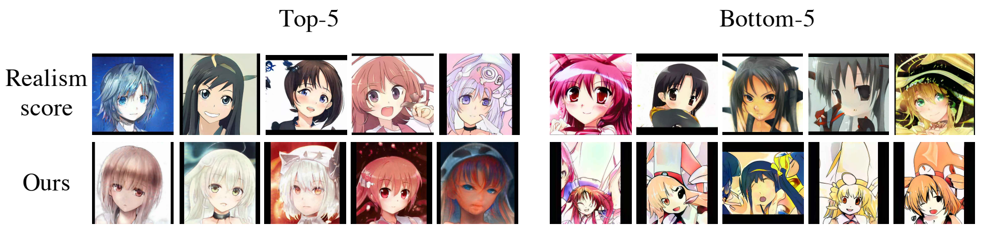


Figure 7.5: Top 5 and Bottom 5 generated 3D shapes for four categories (*i.e.*, airplane, chair, table and rifle) in terms of the proposed latent density score on ShapeNet Core V1 for SDF-StyleGAN. The generated samples with high scores have plausible 3D shapes and complete geometry structures, while samples with low scores exhibit unrealistic shapes and severe geometry distortion.

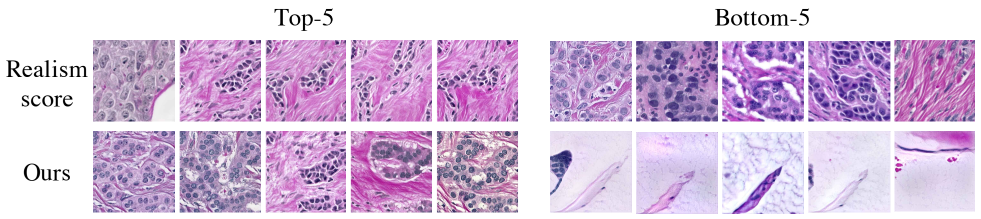
Quality Assessment on Non-ImageNet-like Images

Existing quality assessment methods operate under the assumption that semantically similar images are mapped to points close to each other in the embedding space of a pre-trained feature extractor. However, this assumption might not hold true across different data domains. In this section, we conduct quality assessment for images from two non-ImageNet-like domains, *i.e.*, the medical domain and anime-style domain. Figure 7.6 shows the samples with the highest and lowest latent density scores / realism scores among 5k candidate samples on each domain. We can see that the images with the highest and lowest latent density scores exhibit clear visual difference. On the BreCaHAD dataset [2], for example, the high-density images contain representative human cells while low-density images are mostly blank. We note that the low-density samples do not show degraded perceptual quality. This is probably because although these images are underrepresented cases in the training set, reconstructing them is relatively easy due to their simple layouts.

On the other hand, we do not observe visually distinguishable differ-



(a) StyleGAN2 - Danbooru



(b) StyleGAN2 - BreCaHAD

Figure 7.6: Top and bottom generated images in terms of latent density score and realism score using StyleGAN2 pre-trained on BreCaHAD [2] and Danbooru [15] datasets. There is clear visual difference between images with the highest and lowest latent density scores. In comparison, we do not observe visually distinguishable difference between samples with the highest and lowest realism scores.

ences between samples with the highest and lowest realism scores. This suggests that the pre-trained VGG space used by the realism score is not semantically meaningful for non-ImageNet-like domain images. In addition, our method is more computationally efficient, since we directly operate on the latent codes instead of actually generating all the 5k candidate images.

7.4 Applications

7.4.1 Latent Face Editing

Our proposed method operates directly on the latent space of the generator. Thus, it can be seamlessly incorporated into latent-based image editing methods. Previous work [145] has shown that by moving a latent code along certain directions in the latent space of a well-trained face syn-

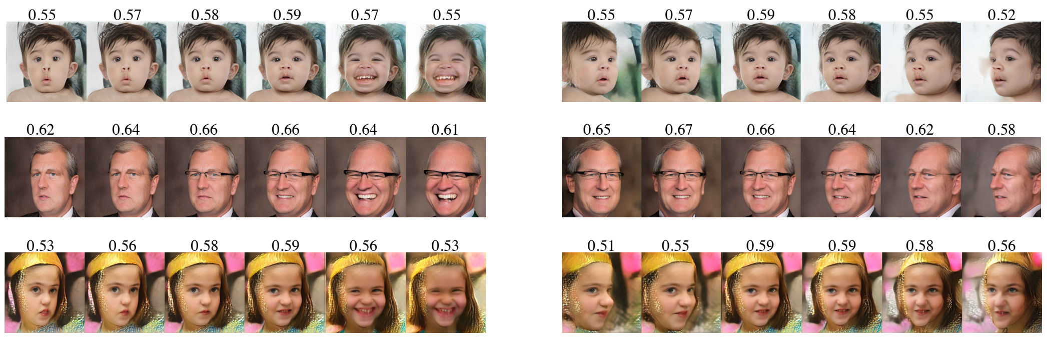


Figure 7.7: Latent-based face editing results and the corresponding latent density scores. The latent density scores highly correlate with the quality of the images.

thesis model, one could control facial attributes of the generated images. However, if the code is moved too far from the well-behaved regions [201] of the latent space, the generated samples will suffer from severe changes [145] as well as degradation in image quality. Here we use our method to estimate the perceptual quality of the edited samples. Specifically, we take a latent code and move it along the direction for the attribute “pose" in the latent space $\mathcal{W}$ of StyleGAN2 following [145]. We compute the latent density score of the moved latent code based on Equation 7.1. Figure 7.7 shows the generated edited images and the corresponding scores.

As shown in the figure, the latent density scores well correlate with the quality of the manipulated images: images with low scores contain artifacts while images with high scores are of better quality. Our method provides a reliable way to assess the quality of edited images even before generating them, which helps to avoid image corruption during latent space traversal and facilitates meaningful image manipulation.

7.4.2 Few-Shot Image Classification

Our method enables selecting strictly high-quality generated images with clear, high-resolution objects. These images are particularly useful for augmenting the training set in low-shot scenarios. Here we show these samples can be used in the task of few-shot image classification and greatly boost performance. Specifically, we synthesize images using a pre-trained text-to-image model, *e.g.*, Stable Diffusion, with the class name as

	Support Samples	*mini*ImageNet 1-shot	*mini*ImageNet 5-shot	*tiered*ImageNet 1-shot	*tiered*ImageNet 5-shot
Real	-	63.17 ± 0.23	79.26 ± 0.17	68.62 ± 0.27	83.74 ± 0.18
SD-generated	bottom-k	63.43 ± 0.45	71.97 ± 0.37	64.62 ± 0.55	75.08 ± 0.45
	random-k	63.87 ± 0.43	72.76 ± 0.38	66.04 ± 0.52	77.10 ± 0.44
	top-k	**67.15 ± 0.44**	**73.60 ± 0.37**	**68.39 ± 0.54**	**77.42 ± 0.43**

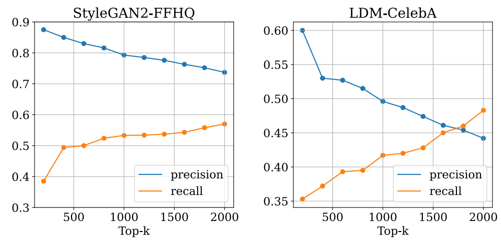
Table 7.1: Few-shot image classification accuracy on *mini*ImageNet and *tiered*ImageNet using images generated from different sets of latent codes. Using images from the latent codes with highest latent density scores achieves better classification performance, which validates the superior quality of images with high latent density scores.

the text condition. The synthesized images are then used as support samples for the corresponding class. We generate k images for k-shot learning ($k = 1$ or 5). For the feature extractor, we use ResNet12 [115] trained following previous work [24]. Table 7.1 compares the performance of using different sets of latent codes during image generation including: 1) k randomly sampled codes, 2) top-k codes with the highest, and 3) top-k codes with the lowest latent density scores. Using samples with high latent density scores as support data leads to better few-shot performance on both the *mini*ImageNet [159] and *tiered*ImageNet [131] datasets for the 1-shot and 5-shot settings. In particular, results on 1-shot *mini*ImageNet show the largest margin, with a 3.28% improvement over using random codes. This validates the superior quality of images generated from codes with high latent density scores.

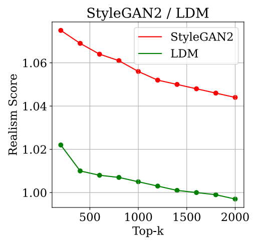
7.5 Analysis

7.5.1 Relationship with Existing Metrics

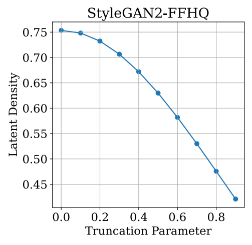
In this section, we investigate the relationship of our proposed metric with other existing metrics. In particular, we generate 2000 fake samples and rank them based on the latent density score. Each time we select top-k samples and calculate the corresponding precision / recall / realism scores of the selected samples. The scores of these metrics under different k values are shown in Figure 7.8 (a) and Figure 7.8 (b). In addition, we show in



(a) Precision / Recall



(b) Realism Score



(c) Truncation

Figure 7.8: Relationship between our proposed latent density score and other metrics. Top-k samples are ranked according to our latent density score. The results of our proposed metric are aligned with existing evaluation metrics on images of common domains.

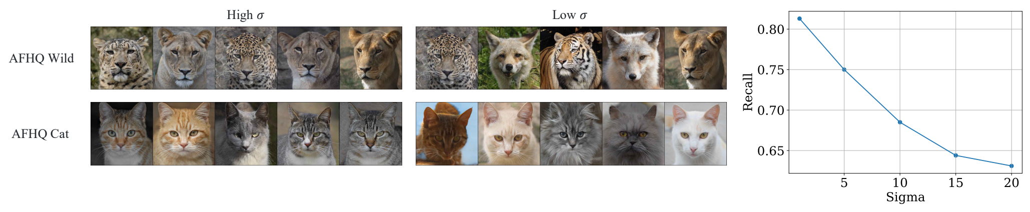
Figure 7.8 (c) how the latent density score changes when we increase the value of truncation parameter used in truncation trick. We conduct this experiment using StyleGAN2 trained on FFHQ [63] and LDM trained on CelebA-HQ [26].

Precision and Recall. Precision and recall are commonly used evaluation metrics in many tasks, such as image classification or natural language processing. In particular, precision measures the fraction of the generated samples that are realistic. Recall, on the other hand, measures the fraction of the real data distribution which can be covered by the distribution of fake data. As shown in Figure 7.8 (a), a small value of k leads to high precision and low recall. This suggests that the samples with high latent density scores are of high quality. As k increases, more diverse samples are selected, which improves recall, while the decrease in precision indicates the newly selected samples are of inferior quality. The correlation between precision / recall and latent density score validates that our proposed metric reliably indicates sample quality.

Realism Score. The realism score is a highly relevant metric that measures the fidelity of an individual generated sample. As shown in Figure 7.8 (b), as k increases, the average realism score of the top-k selected samples decreases for both StyleGAN2 and LDM. This suggests that our proposed metric is aligned with realism score, *i.e.*, samples with low latent density scores also have low realism scores, and vice versa. However, the realism score relies on another feature extractor to project the generated samples to another space. Thus, it is not able to generalize to other domains or modalities (as shown in Section 7.3.2). Moreover, computing realism score requires generating image data, which is time-consuming and not easily scalable, while our method directly operates on the latent space without the need for generating images.

Truncation Trick. Truncation trick [63] is used to increase the fidelity of the generated images in GAN-based generative models by moving the latent code towards the mean latent code. The degree of truncation is controlled by the truncation parameter ψ, *i.e.*, $\psi = 0$ indicates full truncation using the mean code and $\psi = 1$ indicates no truncation. We see from Figure 7.8 (c) that the latent density score decreases as the truncation parameter increases. A higher degree of truncation typically leads to high-fidelity image generation, which, as we show, corresponds to a higher value of the latent density score. This suggests that the latent density score is a valid measure for generated sample quality.

7.5.2 Effect of Hyper-Parameter σ



(a) (b)

Figure 7.9: Images with high latent density scores using high and low σ values on the AFHQ Wild and AFHQ Cat datasets (a) and different recall rates under different σ values (b). We observe that under a lower σ value, the images selected as high-density ones exhibit more diversity, which corresponds to a higher recall rate.

In this section, we analyze how the choice of σ in Equation 7.1 affects quality assessment. Equation 7.1 measures the average Gaussian kernelized Euclidean distance between a given code and the latent codes extracted from training data, with σ being the standard deviation of the kernel function. When applying a small σ, the final density value will rely relatively more on the area surrounding the given code. This will increase the chance that points residing in local clusters are selected as high-density points. As a result, the selected samples are likely to be more diverse. Figure 7.9 (a) shows the images with high latent density scores on the AFHQ Wild and AFHQ Cat datasets [26] under large and small σ values respectively. We observe that the selected high-density samples when using a small σ are more diverse compared to using a large σ. Correspondingly, we observe a higher recall under a smaller σ (as shown in Figure 7.9 (b)), indicating a more complete coverage of the latent manifold. σ enables us to control the relative contribution of local density and global density w.r.t to the final density score. In this case, applying a small σ allows us to select more diverse samples.

7.6 Discussion and Conclusions

We propose a novel approach to estimate sample quality via the latent space of generative models. Our method can be particularly useful in many scenarios. When training generative models, our proposed score points out the underrepresented cases that would possibly require collecting additional data. It also allows us to select high-quality samples that best benefit downstream tasks. For large-scale generative models, pre-generation quality assessment can greatly reduce computational costs. However, only sampling data with high scores, might result in an incomplete coverage of the data manifold. This is because the scores are likely to be higher for large clusters of data representing common cases, as opposed to minority groups such as rare animal species or uncommon medical conditions. One way to alleviate this issue is by considering only small neighborhood areas when measuring the density, which can be achieved by applying a small value of σ. Further, previously proposed sampling techniques such as accept-reject sampling [8] can be used together with our method to increase sample diversity. Combining our score with diversity-related scores also allows us to select diverse samples with high quality. In future work, we intend to extend our method to generative models with high latent dimensions, such as deep hierarchical VAEs [157], or video generative models [173] with an additional temporal dimension in latent space. For these models, latent dimension reduction techniques might be a potential solution.

Chapter 8

Summary and Future Work

8.1 Summary

In this thesis, we investigate how to generate diverse and reliable data for few-shot learning tasks, *i.e.*, few-shot classification, fine-grained few-shot classification, few-shot object detection and class-agnostic object counting. Additionally, we investigate the conditions under which generative models perform well, ensuring the more reliable use of generated samples in FSL.

In Chapter 3, we introduce a VAE-based method to generate reliable data to construct class prototypes for few-shot classification. To achieve this, we propose a sample selection method to collect a set of representative samples for training the VAE model. This training scheme effectively enhances the representativeness of the generated samples and therefore, improves the few-shot classification results.

In Chapter 4, we introduce a method to generate data that represents the distribution of intra-class variance to diversify fine-grained few-shot categories. Specifically, we decompose features into two components, *i.e.*, intra-class variance features and class-discriminative features. We model the intra-class variance features via a common distribution, allowing us to sample variations to diversify unseen instances. This approach enlarges the intra-class variance while preserving the class-discriminative features, thereby benefiting fine-grained few-shot learning.

In Chapter 5, we present a VAE-based method to generate features with increased crop-related diversity for few-shot object detection (FSOD). We first identify the lack of crop-related diversity in few-shot training data as a crucial problem. To resolve this issue, we propose a novel VAE architecture designed to increase crop-related diversity in the generated samples, supporting the training of FSOD classifiers. Our results demonstrate that the increased diversity in the generated samples significantly improves the current state-of-the-art FSOD performance.

In Chapter 6, we present our method to generate reliable class prototypes for zero-shot object counting, a new task where only the class name is available during test time. Starting from a class name, our method can accurately identify the optimal patches which can then be used as counting exemplars. Experimental results on the current class-agnostic counting dataset validate the effectiveness of our method.

Finally, in Chapter 7, we propose a method to assess sample quality by examining the latent space of the generative model. Our approach offers key advantages over previous methods and facilitates downstream FSL tasks.

8.2 Future Work

Throughout the work presented in this thesis, we have demonstrated various use cases of generative models for few-shot learning tasks. Along the research direction of generative modeling, we present three interesting topics for future development.

8.2.1 Diffusion Models for Semantic Segmentation

Text-to-image generative models have shown impressive performance in generating photo-realistic images. One of the main advantages of these models is the strong correspondence between visual pixels and language. This visual-language correspondence is implicitly encoded in the intermediate layers of pre-trained diffusion models and could be highly beneficial for tasks like semantic segmentation if effectively extracted. For example, previous works have shown that the internal cross-attention maps contain rich semantic relationships between the visual objects and the corresponding prompts. If we can infer segmentation masks from these attention maps, we can synthesize an unlimited number of images with pixel-wise pseudo masks for categories within the generative model's vocabulary. These synthesized data, along with the pseudo labels, can serve as a free data source for training segmentation models. This can be particular useful for medical imaging, autonomous driving, and satellite imagery analysis.

8.2.2 Video Editing via Diffusion Models

Video editing using diffusion models presents a compelling direction for future research. Current advancements involve constructing Text-to-Video (T2V) generative models trained on large text-video paired datasets. However, acquiring and annotating these datasets can be prohibitively expensive. A promising approach is to explore self-supervised learning techniques within diffusion models to augment their ability to learn from unlabeled video data. This includes tasks like video inpainting, content manipulation, and adaptation to various video styles or genres, all without the need for extensive labeled datasets. This direction not only reduces dependency on costly annotations but also enhances the versatility and applicability of diffusion models in video editing tasks.

8.2.3 Diffusion Models for Object Detection

Previous work using diffusion models for object detection [22] formulates object detection as a denoising diffusion process from noisy boxes to object boxes. Another potential direction is to leverage their powerful ability to generate high-quality and diverse data to improve object detection. For example, we can create augmented images that introduce variations in lighting, occlusions, and perspectives, thus enhancing the robustness of object detectors. Additionally, diffusion models can denoise and enhance low-quality images, improving the overall quality of the training data, which leads to better feature extraction by the detection algorithms. Furthermore, these models can generate synthetic images that mimic real-world scenarios, providing additional labeled data that enriches the training set. This combination of data augmentation, enhancement, and synthetic data generation makes diffusion models a valuable tool for significantly improving the accuracy and generalizability of object detection systems.

Chapter 9

Bibliography

[1] https://www.hillspet.com/dog-care/dog-breeds/yorkshire-terrier. URL `https://www.hillspet.com/dog-care/dog-breeds/yorkshire-terrier`. 22

[2] A. Aksac, D. J. Demetrick, and T. O. an Reda Alhajj. Brecahad: A dataset for breast cancer histopatholog- ical annotation and diagnosis. *BMC Research Notes*, 2019. 106, 107

[3] A. Alessandro and S. Stefano. Emergence of invariance and disentanglement in deep representations. In *J. Mach. Learn. Res.*, 2018. 44

[4] A. Antoniou, A. Storkey, and H. Edwards. Data augmentation generative adversarial eetworks. In *arXiv preprint arXiv:1711.04340*, 2018. 38

[5] G. Arora, V. K. Verma, A. Mishra, and P. Rai. Generalized zero-shot learning via synthesized examples. In *CVPR*, 2018. 16, 17, 23

[6] C. Arteta, V. S. Lempitsky, J. A. Noble, and A. Zisserman. Interactive object counting. In *ECCV*, 2014. 18

[7] C. Arteta, V. S. Lempitsky, and A. Zisserman. Counting in the wild. In *ECCV*, 2016. 74

[8] S. Azadi, C. Olsson, T. Darrell, I. Goodfellow, and A. Odena. Discriminator rejection sampling. In *ArXiv*, 2019. 112

[9] J. Ba, K. Swersky, S. Fidler, and R. Salakhutdinov. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *ICCV*, 2015. 16, 22

[10] A. Bansal, K. Sikka, G. Sharma, R. Chellappa, and A. Divakaran. Zero-shot object detection. In *ECCV*, 2018. 75

[11] L. Bertinetto, J. F. Henriques, P. Torr, and A. Vedaldi. Meta-learning with differentiable closed-form solvers. *ArXiv*, abs/1805.08136, 2019. 48

[12] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying mmd gans. In *ICLR*, 2018. 19

[13] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao. Yolov4: Optimal speed and accuracy of object detection. *ArXiv*, abs/2004.10934, 2020. 64, 65

[14] A. Borowicz, H. Le, G. Humphries, G. Nehls, C. Höschle, V. Kosarev, and H. Lynch. Aerial-trained deep learning networks for surveying cetaceans from satellite imagery. *PLoS ONE*, 14, 2019. 55

[15] G. Branwen, Anonymous, and D. Community. Danbooru2019 portraits: A large-scale anime head illus- tration dataset. https://www.gwern.net/crops#danbooru2019-portraits. 2019. 107

[16] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019. 100

[17] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding disentangling in β-vae. In *arXiv: Machine Learning*, 2018. 44

[18] Y. Cao, J. Wang, Y. Jin, T. Wu, K. Chen, Z. Liu, and D. Lin. Few-shot object detection via association and discrimination. In *NeurIPS*, 2021. 14, 64

[19] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 98

[20] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 104

[21] R. T. Q. Chen, X. Li, R. Grosse, and D. Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *arXiv preprint arXiv:1802.04942*, 2018. 44

[22] S. Chen, P. Sun, Y. Song, and P. Luo. Diffusiondet: Diffusion model for object detection. In *ICCV*, 2023. 116

[23] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang. A closer look at few-shot classification. In *International Conference on Machine Learning(ICML)*, 2019. 47

[24] Y. Chen, Z. Liu, H. Xu, T. Darrell, and X. Wang. Meta-baseline: Exploring simple meta-learning for few-shot learning. In *ICCV*, 2021. 21, 23, 25, 28, 29, 30, 31, 32, 109

[25] Y.-C. Chen. A tutorial on kernel density estimation and recent advances. *Biostatistics & Epidemiology*, 1:161 – 187, 2017. 34

[26] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha. Stargan v2: Diverse image synthesis for multiple domains. In *CVPR*, 2020. 100, 104, 110, 112

[27] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. In *TPAMI*, 2002. 99

[28] D. L. Davies and D. W. Bouldin. A cluster separation measure. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 1979. 53

[29] D. L. Davies and D. W. Bouldin. Multi-level semantic feature augmentation for one-shot learning. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 1979. 11, 12

[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 29

[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 21

[32] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021. 103

[33] N. Dilokthanakul, P. A. M. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. In *arXiv preprint arXiv:1611.02648*, 2017. 51

[34] Y. Du, J. Xu, X. Zhen, M.-M. Cheng, and L. Shao. Conditional variational image deraining. *IEEE Transactions on Image Processing*, 29: 6288–6301, 2020. 16

[35] N. Dvornik, C. Schmid, and J. Mairal. Diversity with cooperation: Ensemble methods for few-shot classification. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 28

[36] P. Esser, E. Sutter, and B. Ommer. A variational u-net for conditional appearance and shape generation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8857–8866, 2018. 16

[37] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88, 2009. 3, 58, 62

[38] Q. Fan, W. Zhuo, and Y.-W. Tai. Few-shot object detection with attention-rpn and multi-relation detector. *CVPR*, pages 4012–4021, 2020. 14

[39] Q. Fan, W. Zhuo, C.-K. Tang, and Y.-W. Tai. Few-shot object detection with attention-rpn and multi-relation detector. In *CVPR*, June 2020. 13

[40] Q. Fan, C.-K. Tang, and Y.-W. Tai. Few-shot object detection with model calibration. In *ECCV*, 2022. 64

[41] Z. Fan, Y. Ma, Z. Li, and J. Sun. Generalized few-shot object detection without forgetting. In *CVPR*, June 2021. 13, 15

[42] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning(ICML)*, 2017. 9, 28, 47, 48

[43] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. A. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL `https://proceedings.neurips.cc/paper/2013/file/7cce53cf90577442771720a370c3c723-Paper.pdf`. 16

[44] H. Gao, Z. Shou, A. Zareian, H. Zhang, and S.-F. Chang. Low-shot learning via covariance-preserving adversarial augmentation networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 38

[45] S. Gong, S. Zhang, J. Yang, D. Dai, and B. Schiele. Class-agnostic object counting robust to intraclass diversity. In *ECCV*, 2022. 18

[46] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. In *NeurIPS*, 2014. 3, 98

[47] K. Guirguis, A. Hendawy, G. Eskandar, M. Abdelsamad, M. Kayser, and J. Beyerer. Cfa: Constraint-based finetuning approach for generalized few-shot object detection. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4038–4048, 2022. 15

[48] J. Guo and S. Guo. A novel perspective to zero-shot learning: Towards an alignment of manifold structures via semantic feature expansion. *IEEE Transactions on Multimedia*, 23:524–537, 2021. 16, 22

[49] X. Guo, Y. Du, and L. Zhao. Property controllable variational autoencoder via and invertible mutual dependence. In *ICLR*, 2021. 15, 16

[50] G. Han, Y. He, S. Huang, J. Ma, and S.-F. Chang. Query adaptive few-shot object detection with heterogeneous graph convolutional networks. In *ICCV*, October 2021. 13

[51] G. Han, J. Ma, S. Huang, L. Chen, and S.-F. Chang. Few-shot object detection with fully cross-transformer. In *CVPR*, pages 5321–5330, 2022. 13, 64

[52] J. Han, H. Choi, Y. Choi, J. Kim, J.-W. Ha, and J. Choi. Rarity score : A new metric to evaluate the uncommonness of synthesized images. In *ICLR*, 2023. 19, 97, 98

[53] B. Hariharan and R. Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *IEEE International Conference on Computer Vision (ICCV)*, June 2017. 39

[54] N. Hayat, M. Hayat, S. Rahman, S. H. Khan, S. W. Zamir, and F. S. Khan. Synthesizing the unseen for zero-shot object detection. In *ACCV*, 2020. 55

[55] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 30, 36

[56] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, June 2016. 46, 63

[57] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 19, 97, 98

[58] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 15

[59] J. Ho and T. Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 103

[60] R. Hou, H. Chang, B. Ma, S. Shan, and X. Chen. Cross attention network for few-shot classification. In *NeurIPS*, 2019. 28

[61] H. Huang, J. Zhang, J. Zhang, J. Xu, and Q. Wu. Low-rank pairwise alignment bilinear network for few-shot fine-grained image classification. In *arXiv preprint arXiv:1908.01313*, 2019. 48

[62] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell. Few-shot object detection via feature reweighting. *ICCV*, 2019. 14, 62

[63] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 100, 104, 110, 111

[64] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila. Training generative adversarial networks with limited data. In *NeurIPS*, 2020. 101, 104

[65] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 101, 104

[66] P. Kaul, W. Xie, and A. Zisserman. Label, verify, correct: A simple few-shot object detection method. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 13, 14, 15, 64

[67] R. Keshari, R. Singh, and M. Vatsa. Generalized zero-shot learning via over-complete distribution. *CVPR*, pages 13297–13305, 2020. 16, 17, 18

[68] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization(FGVC), IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2011. 38, 40, 46, 47, 48

[69] J. Kim, T.-H. Oh, S. Lee, F. Pan, and I. S. Kweon. Variational prototyping-encoder: One-shot learning with prototypical images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 16, 21

[70] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015. 47

[71] D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018. 100

[72] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014. 3, 40, 98

[73] J. Klys, J. Snell, and R. S. Zemel. Learning latent subspaces in variational autoencoders. In *NeurIPS*, 2018. 15, 16

[74] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila. Mimproved precision and recall metric for assessing generative models. In *NeurIPS*, 2019. 19, 97, 100

[75] H. Le and D. Samaras. Shadow removal via shadow image decomposition. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 38

[76] H. Le and D. Samaras. From shadow segmentation to shadow removal. In *European Conference on Computer Vision(ECCV)*, 2020. 38

[77] H. Le and D. Samaras. Physics-based shadow image decomposition for shadow removal. In *arXiv preprint arXiv:2012.13018*, 2020. 38

[78] H. Le, C.-P. Yu, G. Zelinsky, and D. Samaras. Co-localization with category-consistent features and geodesic distance propagation. In *ICCV Workshop*, 2017. 14

[79] H. Le, T. F. Y. Vicente, V. Nguyen, M. Hoai, and D. Samaras. A+D Net: Training a shadow detector with adversarial shadow attenuation. In *European Conference on Computer Vision(ECCV)*, 2018. 38

[80] H. Le, B. Goncalves, D. Samaras, and H. Lynch. Weakly labeling the antarctic: The penguin colony case. In *CVPR Workshops*, June 2019. 55

[81] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit databases. Technical Report, 2014. 104

[82] J. Lee, H. Kim, Y. Hong, and H. W. Chung. Self-diagnosing gan: Diagnosing underrepresented samples in generative adversarial networks. In *NeurIPS*, 2021. 19

[83] K. Lee, S. Maji, A. Ravichandran, and S. Soatto. Meta-learning with differentiable convex optimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 9, 28, 40, 47

[84] B. Li, B. Yang, C. Liu, F. Liu, R. Ji, and Q. Ye. Beyond max-margin: Class margin equilibrium for few-shot object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 64

[85] K. Li, Y. Zhang, K. Li, and Y. Fu. Adversarial feature hallucination networks for few-shot learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 38

[86] W. Li, L. Wan, J. Xu, J. Huo, Y. Gao, and J. Luo. Revisiting local descriptor based image-toclass measure for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 48

[87] Y. Li, H. Zhu, Y. Cheng, W. Wang, C. S. Teo, C. Xiang, P. Vadakkepat, and T. H. Lee. Few-shot object detection via classification refinement and distractor retreatment. In *CVPR*, June 2021. 14

[88] Z. Li, F. Zhou, F. Chen, and H. Li. Meta-sgd: Learning to learn quickly for few-shot learning. In *arXiv preprint arXiv:1707.09835*, 2017. 9

[89] Y. Lifchitz, Y. Avrithis, S. Picard, and A. Bursuc. Dense classification and implanting for few-shot learning. In *CVPR*, 2019. 9

[90] T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 3, 58, 62

[91] X. Lin, Y. Duan, Q. Dong, J. Lu, and J. Zhou. Deep variational metric learning. In *European Conference on Computer Vision(ECCV)*, 2018. 37, 43

[92] B. Liu, Y. Cao, Y. Lin, Q. Li, Z. Zhang, M. Long, and H. Hu. Negative margin matters: Understanding margin in few-shot classification. In *European Conference on Computer Vision(ECCV)*, 2020. 28

[93] C. Liu, Y. Zhong, A. Zisserman, and W. Xie. Countr: Transformer-based generalised visual counting. In *BMVC*, 2022. 18, 85, 86

[94] J. Liu, L. Song, and Y. Qin. Prototype rectification for few-shot learning. In *ECCV*, 2020. 21

[95] M.-Y. Liu, T. M. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 16

[96] Y. Liu, B. Schiele, and Q. Sun. An ensemble of epoch-wise empirical bayes for few-shot learning. In *European Conference on Computer Vision(ECCV)*, 2020. 21, 23, 25, 28, 29, 30

[97] E. Lu, W. Xie, and A. Zisserman. Class-agnostic counting. In *ACCV*, 2018. 18, 74

[98] W. Luo, X. Yang, X. Mo, Y. Lu, L. S. Davis, J. Li, J. Yang, and S.-N. Lim. Cross-x learning for fine-grained visual categorization. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 48

[99] J. Ma, G. Han, S. Huang, Y. Yang, and S.-F. Chang. Few-shot end-to-end object detection via constantly concentrated encoding across heads. In *ECCV*, 2022. 14, 64

[100] T. Ma, M. Bi, J. Zhang, W. Yuan, Z. Zhang, Y. Xie, S. Ding, and L. Ma. Mutually reinforcing structure with proposal contrastive consistency for few-shot object detection. In *ECCV*, 2022. 14, 64

[101] L. V. D. Maaten and G. E. Hinton. Visualizing data using t-sne. In *Journal of Machine Learning Research*, 2008. 33, 53

[102] A. Majee, K. Agrawal, and A. Subramanian. Few-shot learning for road object detection. *ArXiv*, abs/2101.12543, 2021. 21, 55

[103] A. Majee, A. Subramanian, and K. Agrawal. Meta guided metric learner for overcoming class confusion in few-shot road object detection. *ArXiv*, abs/2110.15074, 2021. 21, 55

[104] P. Mangla, M. K. Singh, A. Sinha, N. Kumari, V. N. Balasubramanian, and B. Krishnamurthy. Charting the right manifold: Manifold mixup for few-shot learning. In *WACV*, pages 2207–2216, 2020. 28

[105] Q. Meng, S. Zhao, Z. Huang, and F. Zhou. Magface: A universal representation for face recognition and quality assessment. *CVPR*, pages 14220–14229, 2021. 72

[106] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *ICLR*, 2013. 35, 63, 65, 66

[107] E. G. Miller, N. E. Matsakis, and P. A. Viola. Learning from one example through shared densities on transforms. *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, 1:464–471 vol.1, 2000. 6

[108] G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38:39–41, 1992. 21, 60

[109] A. Mishra, M. S. K. Reddy, A. Mittal, and H. A. Murthy. A generative model for zero shot learning using conditional variational autoencoders. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2269–22698, 2018. 16, 23

[110] T. N. Mundhenk, G. Konjevod, W. A. Sakla, and K. Boakye. A large contextual dataset for classification, detection and counting of cars with deep learning. In *ECCV*, 2016. 74

[111] T. Munkhdalai, X. Yuan, S. Mehri, and A. Trischler. Rapid adaptation with conditionally shifted neurons. In *International Conference on Machine Learning(ICML)*, 2018. 48

[112] M. F. Naeem, S. J. Oh, Y. Uh, Y. Choi, and J. Yoo. Reliable fidelity and diversity metrics for generative models. In *ICML*, 2020. 19

[113] T. Nguyen, C. Pham, K. Nguyen, and M. Hoai. Few-shot object counting and detection. In *ECCV*, 2022. 18, 85

[114] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv: abs/2112.10741*, 2021. 103

[115] B. Oreshkin, P. R. Loṕez, and A. Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *NeurIPS*, 2018. 109

[116] B. N. Oreshkin, P. R. López, and A. Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *NeurIPS*, 2018. 6, 28

[117] C. Ouyang, C. Biffi, C. Chen, T. Kart, H. Qiu, and D. Rueckert. Self-supervision with superpixels: Training few-shot medical image segmentation without annotation. In *ECCV*, 2020. 21, 55

[118] J. Pan, C. Wang, X. Jia, J. Shao, L. Sheng, J. Yan, and X. Wang. Video generation from single semantic label map. *CVPR*, pages 3728–3737, 2019. 16

[119] Z. Peng, Z. Li, J. Zhang, Y. Li, G.-J. Qi, and J. Tang. Few-shot image recognition with knowledge transfer. In *ICCV*, 2019. 1, 12, 14

[120] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014. 21

[121] J.-M. Perez-Rua, X. Zhu, T. M. Hospedales, and T. Xiang. Incremental few-shot object detection. In *CVPR*, June 2020. 14

[122] L. Qiao, Y. Zhao, Z. Li, X. Qiu, J. Wu, and C. Zhang. Defrcn: Decoupled faster r-cnn for few-shot object detection. *ICCV*, 2021. 14, 56, 58, 59, 63, 64, 65, 66, 67, 68, 69, 70, 71

[123] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning(ICML)*, 2021. 21, 30, 76, 81, 85

[124] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 3, 60, 63, 65, 66, 98

[125] A. Rajeswaran, C. Finn, S. Kakade, and S. Levine. Meta-learning with implicit gradients. In *NeurIPS*, 2019. 9

[126] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text- conditional image generation with clip latents. *ArXiv*, abs/2204.06125, 2022. 97

[127] V. Ranjan and M. Hoai. Exemplar free class agnostic counting. In *ACCV*, 2022. 19, 74, 77, 84, 85, 86, 90

[128] V. Ranjan, U. Sharma, T. Nguyen, and M. Hoai. Learning to count everything. In *CVPR*, 2021. 3, 18, 74, 79, 84, 85, 86, 89

[129] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 9, 10, 29, 48

[130] S. E. Reed, Z. Akata, H. Lee, and B. Schiele. Learning deep representations of fine-grained visual descriptions. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 49–58, 2016. 16

[131] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle, and R. S. Zemel. Meta-learning for semi-supervised few-shot classification. *ArXiv*, abs/1803.00676, 2018. 1, 29, 36, 109

[132] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39, 2015. 55, 63

[133] M. Rezaei and M. Shahidi. Zero-shot learning and its applications from autonomous vehicles to covid-19 diagnosis: A review. *Intelligence-Based Medicine*, 3:100005 – 100005, 2020. 21, 55

[134] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 3, 76

[135] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 97, 98, 100, 101, 102

[136] M. S. M. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly. Assessing generative models via precision and recall. In *NeurIPS*, 2018. 19

[137] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *NeurIPS*, 2016. 19

[138] D. B. Sam, A. Agarwalla, J. Joseph, V. A. Sindagi, R. V. Babu, and V. M. Patel. Completely self-supervised crowd counting via distribution matching. In *ECCV*, 2022. 74

[139] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap. Meta-learning with memory-augmented neural networks. In *ICML*, 2016. 9

[140] E. Schonfeld, S. Ebrahimi, S. Sinha, T. Darrell, and Z. Akata. Generalized zero- and few-shot learning via aligned variational autoencoders. In *CVPR*, 2019. 16

[141] E. Schwartz, L. Karlinsky, J. Shtok, S. Harary, M. Marder, R. Feris, A. Kumar, R. Giryes, and A. M. Bronstein. Delta-encoder: an effective sample synthesis method for few-shot object recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, June 2018. 10, 39, 40, 47, 50, 53

[142] E. Schwartz, L. Karlinsky, J. Shtok, S. Harary, M. Marder, S. Pankanti, R. S. Feris, A. Kumar, R. Giryes, and A. M. Bronstein. Repmet: Representative-based metric learning for classification and few-shot object detection. In *CVPR*, 2019. 14

[143] T. R. Scott, K. Ridgeway, and M. Mozer. Adapted deep embeddings: A synthesis of methods for k-shot inductive transfer learning. In *NeurIPS*, 2018. 9

[144] H. Shao, S. Yao, D. Sun, A. Zhang, S. Liu, D. Liu, J. Wang, and T. F. Abdelzaher. Controlvae: Controllable variational autoencoder. In *ICML*, 2020. 15

[145] Y. Shen, J. Gu, X. Tang, and B. Zhou. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020. 107, 108

[146] M. Shi, H. Lu, C. Feng, C. Liu, and Z. Cao. Represent, compare, and learn: A similarity-aware framework for class-agnostic counting. In *CVPR*, 2022. 18, 74, 79, 85, 86, 89

[147] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2016. 38

[148] C. Simon, P. Koniusz, R. Nock, and M. T. Harandi. Adaptive subspaces for few-shot learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4135–4144, 2020. 28

[149] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 97

[150] J. Snell, K. Swersky, and R. S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 6, 21, 23, 25, 47, 48

[151] K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015. 22, 26, 60

[152] B. Sun, B. Li, S. Cai, Y. Yuan, and C. Zhang. Fsce: Few-shot object detection via contrastive proposal encoding. In *CVPR*, 2021. 14, 55, 64

[153] Q. Sun, Y. Liu, T.-S. Chua, and B. Schiele. Meta-transfer learning for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 47

[154] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2018. 6, 8, 47, 48

[155] Y. Tian, Y. Wang, D. Krishnan, J. Tenenbaum, and P. Isola. Rethinking few-shot image classification: a good embedding is all you need? *ArXiv*, abs/2003.11539, 2020. 6, 28

[156] S. Tsutsui, Y. Fu, and D. Crandall. Meta-Reinforced Synthetic Data for One-Shot Fine-Grained Visual Recognition. In *NeurIPS*, 2019. 38, 46, 50

[157] A. Vahdat and J. Kautz. Nvae: A deep hierarchical variational autoencoder. In *NeurIPS*, 2020. 113

[158] G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 38, 40, 46, 47, 51

[159] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching networks for one shot learning. In *NeurIPS*, 2016. 1, 6, 7, 28, 29, 36, 47, 48, 109

[160] Z. Wan, D. Chen, Y. Li, X. Yan, J. Zhang, Y. Yu, and J. Liao. Transductive zero-shot learning with visual structure constraint. In *NeurIPS*, 2019. 16

[161] W. Wang, Q. Xia, Z. Hu, Z. Yan, Z. Li, Y. Wu, N. Huang, Y. Gao, D. N. Metaxas, and S. Zhang. Few-shot learning by a cascaded framework with shape-constrained pseudo label assessment for whole heart segmentation. *IEEE Transactions on Medical Imaging*, 40:2629–2641, 2021. 21, 55

[162] X. Wang, T. E. Huang, T. Darrell, J. Gonzalez, and F. Yu. Frustratingly simple few-shot object detection. *ArXiv*, abs/2003.06957, 2020. 14, 62, 64

[163] Y. Wang, W.-L. Chao, K. Q. Weinberger, and L. van der Maaten. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *ArXiv*, abs/1911.04623, 2019. 28

[164] Y. Wang, Q. Yao, J. T.-Y. Kwok, and L. M. shuan Ni. Generalizing from a few examples: A survey on few-shot learning. *arXiv: Learning*, 2019. 21

[165] Y.-X. Wang, R. Girshick, M. Hebert, and B. Hariharan. Low-shot learning from imaginary data. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2018. 10, 11

[166] Y.-X. Wang, D. Ramanan, and M. Hebert. Meta-learning to detect rare objects. In *ICCV*, 2019. 14

[167] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010. 21, 38, 40, 46, 47, 48, 50, 51, 55

[168] D. Wertheimer, L. Tang, and B. Hariharan. Few-shot classification with feature map reconstruction networks. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8008–8017, 2021. 23, 28, 30

[169] A. Wu, Y. Han, L. Zhu, and Y. Yang. Universal-prototype enhancing for few-shot object detection. *ICCV*, pages 9547–9556, 2021. 14

[170] A. Wu, S. Zhao, C. Deng, and W. Liu. Generalized and discriminative few-shot object detection via svd-dictionary enhancement. In *NeurIPS*, 2021.

[171] A. Wu, Y. Han, L. Zhu, and Y. Yang. Instance-invariant domain adaptive object detection via progressive disentanglement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:4178–4193, 2022. 14

[172] J. Wu, S. Liu, D. Huang, and Y. Wang. Multi-scale positive sample refinement for few-shot object detection. *ArXiv*, abs/2007.09384, 2020. 14, 64

[173] J. Z. Wu, Y. Ge, X. Wang, S. W. Lei, Y. Gu, Y. Shi, W. Hsu, Y. Shan, X. Qie, and M. Z. Shou. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. In *ICCV*, 2023. 113

[174] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41, 2019. 80

[175] Y. Xian, S. Sharma, B. Schiele, and Z. Akata. F-vaegan-d2: A feature generating framework for any-shot learning. In *CVPR*, 2019. 80

[176] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 104

[177] Y. Xiao and R. Marlet. Few-shot object detection and viewpoint estimation for objects in the wild. In *ECCV*, 2020. 14, 64

[178] W. Xie, J. A. Noble, and A. Zisserman. Microscopy cell counting and detection with fully convolutional regression networks. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, 6, 2018. 74

[179] C. Xing, N. Rostamzadeh, B. N. Oreshkin, and P. H. O. Pinheiro. Adaptive cross-modal few-shot learning. In *NeurIPS*, 2019. 11, 13, 21, 22, 28

[180] J. Xu and H. Le. Generating representative samples for few-shot classification. In *CVPR*, 2022. 61

[181] J. Xu, H. Le, M. Huang, S. Athar, and D. Samaras. Variational feature disentangling for fine-grained few-shot classification. In *Proceedings of the International Conference on Computer Vision*, 2021. 1, 16, 37

[182] X. Yan, Z. Chen, A. Xu, X. Wang, X. Liang, and L. Lin. Meta r-cnn: Towards general solver for instance-level low-shot learning. In *ICCV*, 2019. 14

[183] S. Yang, L. Liu, and M. Xu. Free lunch for few-shot learning: Distribution calibration. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 21

[184] S. Yang, H.-T. Su, W. H. Hsu, and W.-C. Chen. Class-agnostic few-shot object counting. In *WACV*, 2021. 18

[185] Y. Yang, F. Wei, M. Shi, and G. Li. Restoring negative information in few-shot object detection. *ArXiv*, abs/2010.11714, 2020. 14

[186] Z. Yang, Y. Wang, X. Chen, J. Liu, and Y. Qiao. Context-transformer: Tackling object confusion for few-shot detection. In *AAAI*, 2020. 14

[187] H.-J. Ye, H. Hu, D. Zhan, and F. Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8805–8814, 2020. 6, 23, 28, 29, 30

[188] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker. Feature transfer learning for deep face recognition with under-represented data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 39, 43, 51, 52, 53

[189] Z. You, K. Yang, W. Luo, X. Lu, L. Cui, and X. Le. Few-shot object counting with similarity-aware feature enhancement. In *WACV*, 2023. 18, 85, 86

[190] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao. Lsun: construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, 2015. 101

[191] Y. Yu, Z. Ji, J. Han, and Z. Zhang. Episode-based prototype generating network for zero-shot learning. *CVPR*, pages 14032–14041, 2020. 16

[192] B. Zhang, X. Li, Y. Ye, Z. Huang, and L. Zhang. Prototype completion with primitive knowledge for few-shot learning. In *CVPR*, pages 3754–3762, 2021. 21, 22

[193] C. Zhang, Y. Cai, G. Lin, and C. Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12200–12210, 2020. 6

[194] J. Zhang, C. Zhao, B. Ni, M. Xu, and X. Yang. Variational few-shot learning. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 16, 28, 43, 51, 52

[195] L. Zhang, T. Xiang, and S. Gong. Learning a deep embedding model for zero-shot learning. In *CVPR*, pages 3010–3019, 2017. 16, 22

[196] R. ZHANG, T. Che, Z. Ghahramani, and Y. S. Yoshua Bengi and. Metagan: An adversarial approach to few-shot learning. In *NeurIPS*, 2018. 38

[197] W. Zhang and Y.-X. Wang. Hallucination improves few-shot object detection. *CVPR*, pages 13003–13012, 2021. 15, 55, 62, 64

[198] X.-Y. Zheng, Y. Liu, P.-S. Wang, and X. Tong. Sdf-stylegan: Implicit sdf-based stylegan for 3d shape generation. In *SGP*, 2022. 104

[199] Y. Zheng, J. Wu, Y. Qin, F. Zhang, and L. Cui. Zero-shot instance segmentation. In *CVPR*, 2021. 75

[200] C. Zhu, F. Chen, U. Ahmed, Z. Shen, and M. Savvides. Semantic relation reasoning for shot-stable few-shot object detection. In *CVPR*, 2021. 14, 64

[201] P. Zhu, R. Abdal, Y. Qin, J. Femiani, and P. Wonka. Improved stylegan embedding: Where are the good latents? *arXiv preprint arXiv:2012.09036*, 2020. 108

[202] P. Zhu, H. Wang, and V. Saligrama. Don't even look once: Synthesizing features for zero-shot detection. In *CVPR*, 2020. 55, 56

[203] Y. Zhu, C. Liu, and S. Jiang. Multi-attention meta learning for few-shot fine-grained image recognition. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence(IJCAI)*, 2020. 48