CLOSED-FORM SAMPLE PROBING FOR TRAINING GENERATIVE MODELS IN ZERO-SHOT LEARNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

SAMET ÇETİN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

FEBRUARY 2022

Approval of the thesis:

CLOSED-FORM SAMPLE PROBING FOR TRAINING GENERATIVE MODELS IN ZERO-SHOT LEARNING

submitted by **SAMET ÇETİN** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Halit Oğuztüzün
Head of the Department, **Computer Engineering** _____

Assist. Prof. Dr. Ramazan Gökberk Cinbiş
Supervisor, **Computer Engineering, METU** _____

Examining Committee Members:

Assist. Prof. Dr. Emre Akbaş
Computer Engineering, METU _____

Assist. Prof. Dr. Ramazan Gökberk Cinbiş
Computer Engineering, METU _____

Assist. Prof. Dr. Ayşegül Dündar
Computer Engineering, Bilkent University _____

Date: 10.02.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Samet Çetin

Signature :

ABSTRACT

CLOSED-FORM SAMPLE PROBING FOR TRAINING GENERATIVE MODELS IN ZERO-SHOT LEARNING

Çetin, Samet
M.S., Department of Computer Engineering
Supervisor: Assist. Prof. Dr. Ramazan Gökberk Cinbiş

February 2022, 49 pages

Generative modeling based approaches have led to significant advances in generalized zero-shot learning over the past few-years. These approaches typically aim to learn a conditional generator that synthesizes training samples of classes conditioned on class embeddings, such as attribute based class definitions. The final zero-shot learning model can then be obtained by training a supervised classification model over the real and/or synthesized training samples of seen and unseen classes, combined. Therefore, naturally, the generative model ideally needs to produce not only relevant samples, but also those that are sufficiently informative for classifier training purposes. However, existing approaches rely on approximations or heuristics to enforce the generator to produce class-specific samples. In this thesis, we propose a principled approach that shows how to directly maximize the value of training examples for zero-shot model training purposes, by inferring and evaluating the closed-form ZSL models at each generative model training step, which we call sample probing. This approach provides a way to validate the quality of generated samples in an end-to-end manner, where the generator receives feedback directly based on the prediction made on the real samples of unseen classes. Our experimental results show that sample

probing improves the recognition results when integrated into state-of-the-art baselines.

Keywords: generalized zero-shot learning, meta learning, generative models, sample probing

ÖZ

SIFIR ÖRNEKLE ÖĞRENMEDE KAPALI FORM ÖRNEK DEĞERLENDİRME İLE ÜRETİCİ MODEL EĞİTİMİ

Çetin, Samet
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Dr. Öğr. Üyesi. Ramazan Gökberk Cinbiş

Şubat 2022 , 49 sayfa

Son birkaç yılda, üretici modelleme tabanlı yaklaşımlar ile, genelleştirilmiş sıfır örnekle öğrenmede kayda değer ilerlemeler elde edilmiştir. Bu yaklaşımlar tipik olarak nitelik tabanlı tanımlar gibi sınıf gösterimlerine koşullanmış eğitim örneklerini sentezleyen bir koşullu üretici model öğrenmeyi hedeflemektedir. Bu yaklaşımlarda, sıfır örnekle öğrenme modeli, görülmüş ve görülmemiş sınıfların gerçek ve/veya sentezlenmiş eğitim örneklerinin üzerinden bir gözetimli sınıflandırma modeli eğitilmesiyle elde edilebilir. Dolayısıyla, üretici modellerin sınıflandırıcı eğitim amaçlarına uygun olarak tercihen yalnızca alakalı değil aynı zamanda yeterince öğretici örnekler üretmesi gerekmektedir. Fakat varolan yaklaşımlar, üretici modeli sınıfa özgü örnekler ürettirmeye zorlamaya yönelik yakınsamalara veya sezgisel yöntemlere dayanmaktadır. Bu tezde, örnek değerlendirme olarak adlandırdığımız, üretici model eğitiminin her adımında üretilen örnekleri bir kapalı form sıfır örnekle öğrenme modeli aracılığı ile değerlendirmeye tabi tutarak eğitim örneklerinin değerini sıfır örnekle öğrenme amacına yönelik doğrudan maksimize eden ilkeli bir yaklaşım öneriyoruz. Önerilen yaklaşım, bir üretici modelin görülmemiş sınıfların gerçek örnekleri üzerine yapı-

lan öngörüleri baz alan geribeslemeleri doğrudan kullanarak sentezlediği örneklerin kalitelerinin uçtan uca doğrulanabilmesine yönelik bir çözüm sağlamaktadır. Deney sonuçlarımız, kapalı form örnek değerlendirme yaklaşımının en gelişkin referans yöntemlere entegre edildiğinde tanıma sonuçlarını yükselttiğini göstermektedir.

Anahtar Kelimeler: genelleştirilmiş sıfır örnekle öğrenme, meta öğrenme, üretici modeller, örnek değerlendirme

To my beloved family

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my supervisor Asst. Prof. Dr. Ramazan Gökberk Cinbiş for his great support and valuable contributions.

I would like to thank the members of the thesis committee Asst. Prof. Dr. Emre Akbaş and Asst. Prof. Dr. Ayşegül Dündar.

I also thank my friend Orhun Buğra Baran for his valuable collaboration that helped improve parts of this work.

The work in this thesis was supported in part by the TUBITAK Grant 119E597. The numerical calculations reported in this paper were partially performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ConvNet	Convolutional Neural Network
Eq.	Equation
FID	Fréchet Inception Distance
GAN	Generative Adversarial Network
GZSL	Generalized Zero-Shot Learning
VAE	Variational Auto Encoder
ZSL	Zero-Shot Learning

CHAPTER 1

INTRODUCTION

State-of-the-art works in various vision problems heavily rely on supervised training. However, supervised training is generally considered inconvenient and expensive, requiring the collection of a large amount of data and annotation. Although there are a lot of freely available unstructured data sources, most of the time, it is hard to collect structured data specific to the vision problem at hand. Additionally, the laborious task of data annotation is time-consuming, error-prone, and generally requires expertise in fine-grained problems. To reduce the annotation overhead, various approaches are proposed such as few-shot learning [1, 2, 3], unsupervised pretraining [4, 5], semi-supervised learning [6] etc. Among these, *Zero-shot Learning* (ZSL) has recently received great interest for being one of the promising paradigms towards building very large vocabulary (visual) understanding models with limited training data.

The problem of ZSL can be summarized as the task of transferring information across classes such that the instances of *unseen* classes, with no training examples, can be recognized at test time, based on the training samples of *seen* classes. To be able to achieve this knowledge transfer, an auxiliary information source representing seen and unseen classes together in the same semantic space, such as manually crafted attribute vectors indicating a common attribute (shared among all classes) in each dimension, hyper-dimensional word vectors automatically extracted from language models, etc. is defined. The idea is to learn a mapping from the visual space to the semantic space at the training phase and make accurate classifications at the test phase. However, in the ZSL setting, the classifier is enforced to make predictions among only unseen classes at test time. This is not practical since the ultimate goal is to generalize well to any target class with minimum or no restrictions by learning

from the limited training data.

Generalized Zero-Shot Learning (GZSL) [7, 8] is introduced soon after and used to refer to a practically more valuable variant of ZSL where both seen and unseen classes may appear at test time. GZSL brings in additional challenges since GZSL models need to produce confidence scores that are comparable across all classes.

In this thesis, we primarily investigate the generalized zero-shot learning problem by exploiting closed-form zero-shot learning models with exact solutions in generative model training. Before we elaborate on the details of our work, in this chapter, we first present the overview of the zero-shot learning problem and our contributions to the literature.

1.1 Overview

Earlier work focuses on discriminative training of ZSL models, such as those based on bilinear compatibility functions [9, 10]. However, such models tend to yield higher confidence scores towards seen classes. As a result, although the models have relatively successful performance in the ZSL setting, they perform very poorly in the GZSL setting.

Recent work shows that hallucinating unseen class samples through statistical generative models can be an effective strategy, *e.g.* [11, 12, 13, 14, 15, 16, 17, 18], reducing the GZSL problem into a supervised classification problem. These approaches rely on generative models conditioned on *class embeddings*, obtained from auxiliary semantic knowledge, such as visual attributes [13], class name word embeddings [16], or textual descriptions [15]. The resulting synthetic examples, typically in combination with existing real examples, are used for training a supervised classifier. By design, the conditional generative model training formulation plays a critical role in the success of the resulting models. Naturally, the resulting models produce comparable scores across seen and unseen classes.

In generative GZSL approaches, the *quality* of class-conditional samples is crucial for building accurate recognition models. It is not straightforward to formally define

the criteria of *good training samples*. Arguably, however, samples need to be (i) realistic (*e.g*. free from unwanted artifacts), (ii) relevant (*i.e*. belong to the desired class distribution) and (iii) informative (*i.e*. contain examples defining class boundaries) to train an accurate classifier. Clearly, a primary factor affecting the quality of generated samples is the loss driving the conditional generative model training process.

There are a few recent works that utilize auxiliary components such as a decoder [18], a refinement module [19], and a normalization scheme [20] to enforce the generator to improve synthetic feature quality and synthesize more discriminative and semantically consistent features. However, none of these approaches directly measure the quality of the synthesized features for training classification models. The fundamental challenge here is the back-propagation over long compute chains, which is expensive and prone to gradient vanishing.

1.2 Contributions

In this thesis, we aim to address the problem of training data generating models via an end-to-end mechanism that we call *sample probing*. [1] Our main goal is to *directly* evaluate the ability of a generative model in synthesizing training examples. To this end, we observe that we can leverage classification models with closed-form solvers to efficiently measure the quality of training samples, in an end-to-end manner. More specifically, we formulate a simple yet powerful meta-learning approach: at each training iteration, (i) take a set of samples from the generative model for a randomly selected subset of classes, (ii) train a zero-shot *probing model* using only the synthesized samples, and (iii) evaluate the probing model on real samples from the training set. We then use the loss value as an end-to-end training signal for updating the generative model parameters at every training iteration. While one can use an arbitrary classifier, we specifically focus on probing models with exact closed-form solutions since it is not efficient or even feasible to back-propagate over a long compute chain. Optimization of the probing models with exact closed-form solutions can be simplified into a differentiable linear algebraic expression and took part as a

[1] Our use of the *sample probing* term is not closely related to the natural language model analysis technique known as *probing* [21, 22, 23].

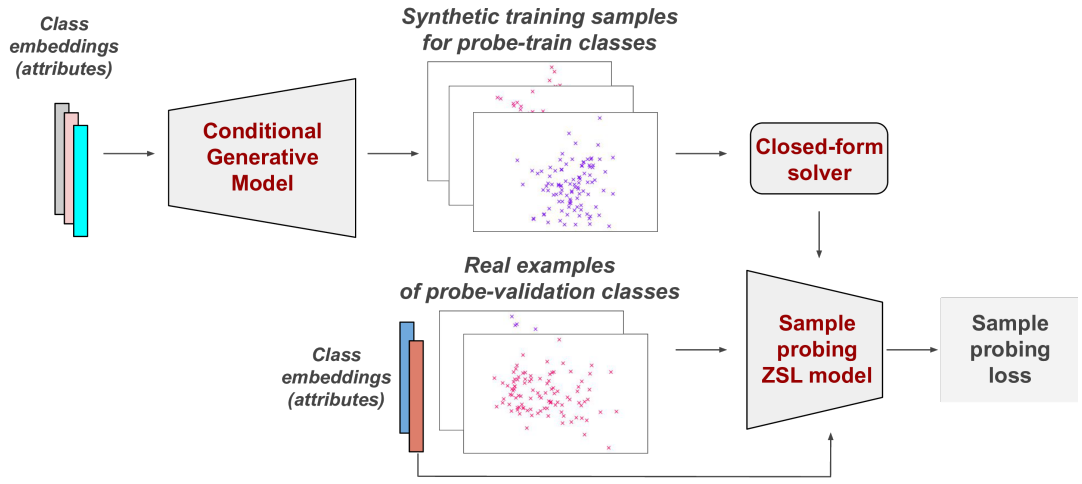

Figure 1.1: Illustration of the proposed framework for the end-to-end *sample probing* of conditional generative models. At each training iteration, we take synthetic training examples for some subset of seen classes (*probe-train classes*) from the conditional generative models, train a closed-form solvable zero-shot learning model (*sample probing ZSL model*) over them and evaluate it on the real examples of a different subset of seen classes (*probe-validation classes*). The resulting cross-entropy loss of the probing model is used as a loss term for the generative model update.

differentiable unit within the compute graph. A graphical summary of the proposed training scheme is given in Figure 1.1. Our quantitative and qualitative experimental results show that the proposed sample probing scheme increases the overall synthetic feature quality and further improves the performance of competitive baseline generative ZSL models under different configurations.

1.3 Outline

In the rest of this thesis, we present an overview of the related work in Chapter 2. In Chapter 3, we first formally define the generalized zero-shot learning problem and then define a mathematical framework to summarize the core training dynamics of mainstream generative GZSL approaches and express our approach in the context of this mathematical framework. In Chapter 4, we present the details of our experimental setup and a thorough experimental evaluation on widely used GZSL benchmark

datasets in which the results show that sample probing yields improvements when introduced into state-of-the-art baselines. We conclude with final remarks in Chapter 5.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we present recent and related work to our proposed approach. We then discuss the details of generative zero-shot learning models that we use as baselines throughout our experiments.

2.1 Related work

The generalized zero-shot learning problem has been introduced by [7] and [24]. The extensive study in [8] has shown that the success of methods can greatly vary across zero-shot and generalized zero-shot learning problems. The additional challenge in the generalized case is the need for deciding whether an input test sample belongs to a seen or unseen class. Discriminative training of ZSL models, such as those based on bilinear compatibility functions [9, 10], are likely to yield higher confidence scores for seen classes. To alleviate this problem, a few recent works have proposed ways to regularize discriminative models towards producing comparable confidence scores across all classes and avoid over-fitting, *e.g.* [25, 26, 27]. For example, [25] estimates correspondence between unseen and seen classes to define an unseen class aware training loss. Similarly, [26] applies entropy-based regularization on unseen class scores.

Generative approaches to zero-shot learning naturally address the confidence score calibration problem. However, in general, generative modeling corresponds to a more sophisticated task than learning discriminant functions only [28]. In this context, the problem is further complicated by the need of predicting zero-shot class distributions. To tackle this challenging task, a variety of techniques have been pro-

posed [11, 12, 13, 14, 15, 16, 17, 29, 18] by adapting generative models, such as VAEs [30, 31] and GANs [32]. To enforce class conditioning, the state-of-the-art approaches use a mixture of heuristics: [12] uses the loss of a pre-trained classifier over the generated samples during training. [18] additionally uses the loss of a sample-to-attribute regressor, in combination with a feedback mechanism motivated from *feedback networks* [33]. [20] introduces *soul samples* which are the average representations of classes to bring a normalization effect on generator training. [19] uses a *feature refinement* module together with a *self-adaptive margin center loss* to enforce the generator to learn discriminative class relevant features. [14] uses *projection discriminator* [34] and *gradient matching loss* as a gradient-based similarity measure for comparing real versus synthetic data distributions. None of these generative ZSL approaches, however, directly measure the *value* of the generated samples *for training* classification models. The main difficulty lies in the need for back-propagating over long compute chains, which is both inefficient and prone to gradient vanishing problems. To the best of our knowledge, we introduce the first end-to-end solution to this problem by the idea of using probing models with closed-form solvers to monitor the sample quality for model training purposes.

Our approach is effectively a *meta-learning* [35] scheme. Meta-learning is a prominent idea in few-shot learning, where the goal is learning to build predictive models from a limited number of samples. The main motivation is the idea that general-purpose classification models may behave sub-optimally when only a few training samples are provided. A variety of meta-learning driven few-shot learning models have been proposed, such as meta-models that transform a few samples to classifiers [36, 35], set-to-set transformations for improving sample representations [37, 38], fast adaptation networks for a few examples [1, 39, 40]. In contrast to such mainstream *learning to classify* and *learning to adapt* approaches, we aim to address the problem of *learning to generate training examples* for GZSL.

There are only a few recent studies that aim to tackle generative ZSL via meta-learning principles. [41, 42] embrace the learning-to-adapt framework of MAML [1], which originally aims to learn the optimal initialization for few-shot adaptation. The MAML steps are incorporated by iteratively applying a single-step update to the generative model using the generative model (VAE/GAN) loss terms, and then back-

propagating over the re-computed generative loss terms on new samples from a disjoint subset of classes using the single-step updated model. Our approach differs fundamentally, as we propose to use the discriminative guidance of ZSL models fully-trained directly from generated sample batches. In another work, [43] proposes an *episodic training*-like approach with periodically altered training sets and losses during training, to learn non-stochastic mappings between the class embeddings and class centers. [44], similarly inspired from meta-learning and mixup regularization [45], proposes to train a novel discriminative ZSL model over episodically defined virtual training classes obtained by linearly mixing classes. Neither of these approaches learns sample generating models, therefore, they have no direct relation to our work focusing on the problem of measuring the sample quality for ZSL model training purposes, with end-to-end discriminative guidance.

The use of recognition models with closed-form solvers has attracted prior interest in various contexts. Notably, [46] proposes the *Embarrassingly Simple Zero-shot Learning* (ESZSL) model as a simple and effective ZSL formulation. We leverage the closed-form solvability of the ESZSL model as part of our approach. [47] utilizes ridge-regression based task-specific few-shot learners within a discriminative meta-learning framework. In a similar fashion, [48, 49] tackle the problem of *video object segmentation* (VOS) and use ridge-regression-based task-specific segmentation models within a meta-learning framework. None of these approaches aim to use recognition models with closed-form solvers to form guidance for generative model training.

Another related research topic is generative few-shot learning (FSL), where the goal is learning to synthesize additional in-class samples from a few examples, *e.g.* [50, 51, 52, 53, 54]. Among these, [51] is particularly related to following a similar motivation of learning to generate good training examples. This is realized by feeding generated samples to a meta-learning model to obtain a classifier, apply it to real query samples, and use its query loss to update the generative model. Apart from the main difference in the problem definition (GZSL vs FSL), our work differs mainly by fully training a closed-form solvable ZSL model from scratch at each training step, instead of a few-shot meta-learners that are jointly trained progressively with the generative model. For example, in [50] the aim is to learn transformation function which

can be used on a few examples to obtain transformed new examples that can be used during classifier training. Newer approaches based on GANs and VAEs learn to synthesize new examples based on prior knowledge coming from examples at hand, *e.g.* [55, 16]. Most of these approaches also train the feature extraction backbone through meta-learning as well.

Finally, we note that meta-learning-based approaches have also been proposed for a variety of different problems, such as learning-to-optimize [56, 57] and long-tail classification [58, 59]. In contrast to these meta-learning approaches, our main goal is to directly measure the quality of the generative model by end-to-end fitting zero-shot models purely using the synthesized samples and back-propagating their zero-shot recognition loss for the generative model.

2.2 Background

Before we dive into the details of the sample probing approach in the following chapter, here we present overviews of the baseline generative zero-shot learning models into which the proposed sample probing scheme is integrated throughout the experiments.

2.2.1 cWGAN

cWGAN [34] is the simplest one among all generative ZSL baseline models that we integrated the sample probing scheme into and performed experiments with. cWGAN consists of a generator G which synthesizes fake features from random noise z and class embedding a, and a discriminator D tries to discriminate real features x and generated features $G(z, a)$. Both G and D are conditioned on the class embedding a. Optimized loss is given by,

$$L_{wgan} = \mathbb{E}\left[D(x, a)\right] - \mathbb{E}\left[D(G(z, a), a)\right] - \lambda\, \mathbb{E}\left[(\|\Delta D(G(z, a), a)\|_2 - 1)^2\right] \quad (2.1)$$

where λ is the penalty coefficient.

2.2.2 TF-VAEGAN

TF-VAEGAN [18] achieves state-of-the-art results in GZSL setting by introducing a *semantic embedding decoder* (SED) into an already strong feature generating VAE-GAN (f-VAEGAN) [13] consisting of f-VAE and f-WGAN. f-VAE consists of an encoder E conditioned on class embedding, that learns a mapping from feature space to latent space, and a decoder G (shared between f-VAE and f-WGAN) conditioned on class embedding that reconstructs image feature x from a random noise z. Optimized loss is given by,

$$L_{vae} = \mathrm{KL}(E(x,a)\|p(z|a)) - \underset{E(x,a)}{\mathbb{E}}[\log G(z,a)] \tag{2.2}$$

where KL is the Kullback-Leibler divergence, $p(z,a)$ is a prior distribution and $\log G(z,a)$ is the reconstruction loss.

The generator of f-WGAN, generates a feature x from noise z and the discriminator D tries to discriminate real and fake features. Both G and D are conditioned on class embeddings a. Optimized loss is defined in Eq. (2.1). Hence, the f-VAEGAN [13] is then optimized by,

$$L_{vaegan} = L_{vae} + \alpha L_{wgan} \tag{2.3}$$

where α is the weight of WGAN loss that needs to be tuned. Semantic embedding decoder Dec, reconstructs class embeddings a from the synthesized features $G(z,a)$ and is optimized using l_1 reconstruction loss by:

$$L_R = \mathbb{E}\left[\|Dec(x) - a\|_1\right] + \mathbb{E}\left[\|Dec(G(z,a)) - a\|_1\right] \tag{2.4}$$

Hence, the final TF-VAEGAN loss formulation of is then defined as,

$$L_{tfvaegan} = L_{vaegan} + \beta L_R \tag{2.5}$$

where β is a hyper-parameter for semantic embedding decoder reconstruction error weighting. Additionally, TF-VAEGAN uses a feedback mechanism that we avoid in our experiments for simplicity as we discuss in Chapter 4.

2.2.3 LisGAN

LisGAN [20] generates synthetic image features through a conditional WGAN. Generator synthesize fake features from a random noise z, and class embedding a, whereas

the discriminator D tries to effectively discriminate real x and fake features $G(z, a)$. Loss formulation of generator G is given by,

$$L_G = -\mathbb{E}\left[D(G(z,a))\right] - \lambda\,\mathbb{E}\left[(\log P(y|G(z,a))\right] \tag{2.6}$$

where the first two terms are the Wasserstein loss and the classification loss on the synthesized feature respectively. λ is a weighting parameter. Loss formulation of discriminator D is given by,

$$\begin{aligned} L_D = {} & \mathbb{E}\left[D(G(z,a))\right] - \mathbb{E}\left[D(x)\right] \\ & - \lambda(\mathbb{E}\left[(\log P(y|G(z,a))\right] + \mathbb{E}\left[\log P(y|x)\right]) \\ & - \beta\,\mathbb{E}\left[(\|\Delta D(G(z,a))\|_2 - 1)^2\right] \end{aligned} \tag{2.7}$$

where β is a hyper-parameter. The last three terms are the classification loss on synthetic samples, the classification loss on real samples, and the enforcer of the Lipschitz constraint, respectively. Additionally, [20] introduces the concept of *soul samples* which are the average representations of classes, a similar idea to prototypical networks for FSL [35], to enhance the general quality of the synthetic features by regularizing the generator training.

2.2.4 FREE

FREE [19], similar to TFVAEGAN [18], base their approach on f-VAEGAN [13], consisting of a f-VAEGAN and a feature refinement module. f-VAEGAN aims to learn a mapping from semantic space to visual space for feature generation by optimizing Eq. (2.3) as already discussed in Section 2.2.2. Additionally, to learn discriminative feature representations a feature refinement module constrained by the *self-adaptive margin center loss* (SAMC-loss) which encourages intra-class compactness and inter-class separability and a *semantic cycle consistency loss* that pushes the feature refinement module to learn semantically-relevant representations, are introduced. Overall objective consisting of the jointly trained encoder E, generator G, discriminator D, and feature refinement module FR can be formulated as,

$$L_{FREE}(E, G, D, FR) = L_{wgan} + L_{vae} + \lambda_{samc} L_{samc} + \lambda_{scc} L_{scc} \tag{2.8}$$

where L_{wgan} and L_{vae} are the f-VAEGAN loss components defined in Eq. (2.1) and Eq. (2.2) respectively; L_{samc} and L_{scc} indicating the self-adaptive margin center loss

and the semantic cycle-consistency loss and, λ_{samc} and λ_{scc} are the weights controlling corresponding losses.

In Chapter 4, we present experimental results showing the performance of all aforementioned competitive generative ZSL baselines on four benchmark GZSL datasets, with and without the proposed sample probing scheme.

CHAPTER 3

METHOD

In this chapter, we first formally define the generalized zero-shot learning problem and then define a mathematical framework to summarize the core training dynamics of mainstream generative GZSL approaches. We then express our approach in the context of this mathematical framework.

3.1 Problem definition

In zero-shot learning, the goal is to learn a classification model that can recognize the test instances of *unseen* classes $\mathcal{Y}_u$, which has no training examples, based on the model learned over the training examples provided for the disjoint set of *seen* classes $\mathcal{Y}_s$. We refer to the class-limited training set by $\mathcal{D}_{\text{tr}}$, which consists of sample and class label pairs $(x \in \mathcal{X}, y \in \mathcal{Y}_s)$. In our work, we focus on ZSL models where $\mathcal{X}$ is the space of image representations extracted using a pre-trained ConvNet. In generalized zero-shot learning, the goal is to build the classification model using the training data set $\mathcal{D}_{\text{tr}}$, such that the model can recognize both seen and unseen class samples at test time. For simplicity, we restrict our discussion to the GZSL problem setting below.

In order to enable the recognition of unseen class instances, it is necessary to have visually-relevant prior knowledge about classes so that classes can visually be related to each other. Such prior knowledge is delivered by the mapping $\psi : \mathcal{Y} \rightarrow \mathcal{A}$, where $\mathcal{A}$ expresses the prior knowledge space. In most cases, the prior knowledge is provided as d_ψ-dimensional vector-space embeddings of classes, obtained using visual attributes, taxonomies, class names combined with word embedding models,

the textual descriptions of classes combined with language models, see *e.g.* [60]. Following the common terminology, we refer to ψ as the *class embedding* function.

3.2 Generative GZSL

In our work, we focus on generative approaches to GZSL. The main goal is to learn a conditional generative model $G : \mathcal{A} \times \mathcal{Z} \rightarrow \mathcal{X}$, which takes some class embedding $a \in \mathcal{A}$ and stochasticity-inducing noise input z, and yields a synthetic sample $x \in \mathcal{X}$. Once such a generative model is learned, synthetic training examples for all classes can simply be sampled from the G-induced distribution P_G, and the final classifier over $\mathcal{Y}$ can be obtained using any standard supervised classification model. We refer to the trainable parameters of the model G by θ_G.

As summarized in Chapter 2, existing approaches vary greatly in terms of their generative model details. For the purposes of our presentation, most of the GZSL works (if not all) can be summarized as the iterative minimization of some loss function that acts on the outputs of the generative model:

$$L_G = \mathop{\mathbb{E}}_{(x,a)\sim\mathcal{D}_{\text{tr}}} \left[\ell_G \left(G(a, z_x), a\right)\right] \tag{3.1}$$

where z_x refers to the noise input associated with the training sample (x, y) and ℓ_G is the generative model learning loss. $(x, a) \sim \mathcal{D}_{\text{tr}}$ is shorthand notation for $(x, \psi(y)) \sim \mathcal{D}_{\text{tr}}$. At each iteration the goal is to reduce L_G approximated over a mini-batch of real samples and their class embeddings.

In our notation, we deliberately keep certain details simple. Noticeably, z_x greatly varies across models. For example, in the case of a conditional GAN model, $z_x \sim p(z)$ can simply be a sample from a simple prior distribution $p(z)$, *e.g.* as in [12, 14, 29]. In contrast, in variational training, z_x is the latent code sampled from a variational posterior, *i.e.* $z_x \sim q(z|x)$, where the variational posterior $q(z|x)$ is given by a variational encoder trained jointly with G, *e.g.* as in [18, 13].

Another important simplification that we intentionally make in Eq. (3.1) is the fact that we define the generative model learning loss ℓ_G as a function of generator output and class embedding, to emphasize its sample-realisticity and class-relevance estima-

tion goals. However, the exact domain of ℓ_G heavily depends on its details, which typically consists of multiple terms and/or (adversarially) trained models. In most of the state-of-the-art approaches, this term is a combination of VAE reconstruction loss [13], conditional or unconditional adversarial discriminator network [12, 18, 14, 13], a sample-to-class classifier for measuring class relevance [12] and sample-to-embedding mappings [18]. The loss L_G may also incorporate additional regularization terms, such as ℓ_2 regularization or a gradient penalty term [14].

In the following, we explain our sample probing approach as a loss term that can, in principle, be used in conjunction with virtually any of the mainstream generative zero-shot learning formulations.

3.3 Sample probing as generative model guidance

The problem that we aim to address is enforcement of G to learn to produce samples maximally beneficial for zero-shot model training purposes. We approach this problem through a *learning to generate training samples* perspective, where we aim to monitor the quality of the generative model through the synthetic class samples it provides.

In construction of our approach, at each training iteration t, we first randomly select a subset of $Y^t_{\text{pb-tr}} \subset \mathcal{Y}$ of seen classes. We refer to these classes as *probe-train* classes. This subset defines the set of classes that are used for training the iteration-specific *probing model* over the synthetic samples. More specifically, we first take samples from the model G with the parameters θ^t_G for these classes, and fully train a temporary ZSL model over them using regularized loss minimization:

$$\Gamma^t = \arg\min_{\Gamma} \mathop{\mathbb{E}}_{x=G(z,a\sim A^t_{\text{pb-tr}})} \left[\ell_{\text{pb}}\left(f_{\text{pb}}(x,a),a\right)\right] \tag{3.2}$$

where f_{pb} is the scoring function of the temporary probing model parameterized by Γ and ℓ_{pb} is its training loss. $A^t_{\text{pb-tr}}$ is the set of class embeddings of classes in $Y^t_{\text{pb-tr}}$. Regularization term over Γ is not shown explicitly for brevity.

The result of Eq. (3.2), gives us a purely synthetic sample driven model Γ^t, which we leverage as a way to estimate the success of the generator in synthesizing training

examples. For this purpose, we sample real examples from the training set $\mathcal{D}_{\text{tr}}$ as validation examples for the probe model. Since we use a (G)ZSL model as the probing model, we can evaluate the model on examples of the classes not used for training the model. Therefore, we sample these probe-validation examples from the remaining classes $Y_{\text{pb-val}} = \mathcal{Y}_s \setminus Y^t_{\text{pb-tr}}$, *i.e.* the classes with real training examples but unused for probe model training, and use softmax cross-entropy loss over these samples as the probing loss:

$$L_{\text{pb}} = - \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_{\text{pb-val}}} \left[\log p(y|x;\Gamma^t)\right] \tag{3.3}$$

where $\mathcal{D}_{\text{pb-val}} \subset \mathcal{D}_{\text{tr}}$ is the data subset of classes $Y_{\text{pb-val}}$. $p(y|x;\Gamma^t)$ is the target class likelihood obtained by applying softmax to $f_{\text{pb}}(x, \psi(y); \Gamma^t)$ scores over the set of target class set. Here, as target class set, one can use only the classes in $Y_{\text{pb-val}}$ (ZSL probing) or those in both $Y_{\text{pb-tr}}$ and $Y_{\text{pb-val}}$ (GZSL probing). We treat this decision as a hyper-parameter and tune on the validation set.

We use a weighted combination of L_{pb} and L_G, as our final loss function. Therefore, the gradients $\nabla_{\theta_G} L_{\text{pb}}$ act effectively as the training signal for guiding G towards yielding training examples that results in (G)ZSL probing models with minimal empirical loss.

3.4 Closed-form probe model

A critical part of construction is the need for a probing model where minimization of Eq. (3.2) is both efficient and differentiable, so that the solver itself can be a part of the compute graph. Probing models that require iterative gradient descent based optimization are unlikely to be suitable as one would need to make a large number of probing model updates for each single G update step, which is both inefficient and prone to gradient vanishing problems. We address this problem through the use of a ZSL model that can be efficiently fit using a closed-form solution.

For this purpose, we opt to use the ESZSL [46] as the main closed-form probe model in our experiments. The model is formalized by the following minimization problem:

$$\min_{\Gamma} \|X^{\text{T}}\Gamma A - Y\|^2_{\text{Fro}} + \Omega\left(\Gamma\right) \tag{3.4}$$

where $X \in R^{d_x \times m}$ and $A \in R^{d_\psi \times k}$ represent the feature and class embeddings corresponding to m input training examples and k classes, Y is the $\{0,1\}^{m \times k}$ matrix of groundtruth labels, $\Gamma \in R^{d_x \times d_\psi}$ is the compatibility model matrix, and $\Omega(\Gamma)$ is the regularization function, defined as:

$$\Omega(\Gamma) = \lambda_x \|\Gamma A\|_{\text{Fro}}^2 + \lambda_a \|X^{\text{T}}\Gamma\|_{\text{Fro}}^2 + \lambda_n \|\Gamma\|_{\text{Fro}}^2 \tag{3.5}$$

where $\lambda_x, \lambda_a, \lambda_n$ correspond to term weights. When the regularization term weights are set such that $\lambda_n = \lambda_x \lambda_a$, the optimal solution to Eq. (3.4) can be computed in a closed-form:

$$\Gamma^\star = (XX^{\text{T}} + \lambda_x I)^{-1} XYA^{\text{T}}(AA^{\text{T}} + \lambda_a I)^{-1} \tag{3.6}$$

This approach was originally proposed as a standalone label-embedding based ZSL model in [46], with the practical advantage of having an efficient solver. Here, we re-purpose this approach as a probing model in our framework, where the fact that the model is solvable in closed-form is critically important, enabling the idea of end-to-end sample probing. For this purpose, we utilize the solver given by Eq. (3.6) as the implementation of Eq. (3.2), which takes a set of synthetic training samples and estimates the corresponding probing model parameters.

3.5 Alternative probe models

While we utilize ESZSL in our main experiments, we demonstrate the possibility of using the proposed approach with different probe models using two additional alternatives. The first one, which we call *Vis2Sem*, is the regression model from visual features to their corresponding class embeddings, defined as follows (using the same notation as in ESZSL):

$$\min_{\Gamma} \|\Gamma^{\text{T}} X - AY^{\text{T}}\|_{\text{Fro}}^2 + \lambda_n \|\Gamma\|_{\text{Fro}}^2. \tag{3.7}$$

A discussion of the Vis2Sem model can be found in [61]. The second one, which we call *Sem2Vis*, is the class embeddings to visual features regression model of [62]:

$$\min_{\Gamma} \|X - \Gamma AY^{\text{T}}\|_{\text{Fro}}^2 + \lambda_n \|\Gamma\|_{\text{Fro}}^2. \tag{3.8}$$

Both models, just like ESZSL, are originally defined as non-generative ZSL models, and we re-purpose them to define our data-dependent generative model training

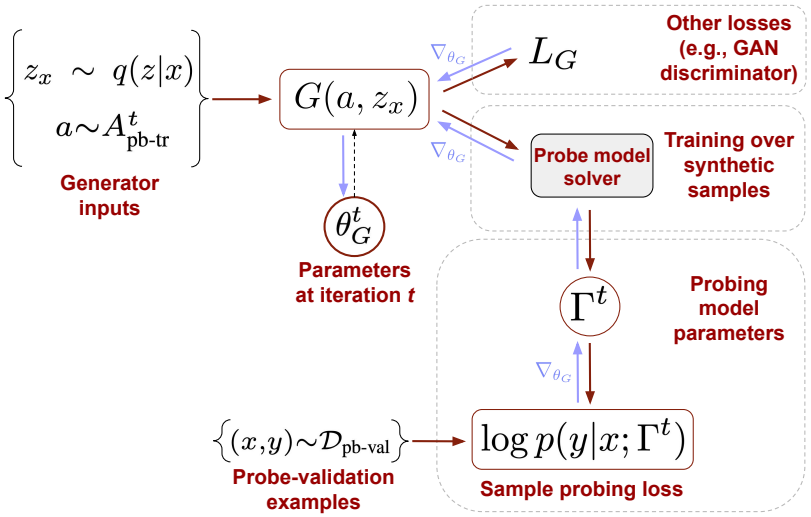

Figure 3.1: The compute graph view of the proposed approach, at some training iteration t. The upper half shows sampling from the generative model and the lower half shows sample probing model loss. Circles denote generator and probing model parameters. Blue arrows show the back-propagation path for updating the generative model. Best viewed in color.

losses. Unlike the bi-linear compatibility model of ESZSL, however, these models rely on distance based classification, and do not directly yield class probability estimates. While one can still obtain a probability distribution over classes, *e.g.* by applying softmax to negative ℓ_2 distances, for simplicity, we directly use the Sem2Vis and Vis2Sem based distance predictions between the visual features of probe-validation samples and their corresponding class embeddings to compute L_{pb} as a replacement of Eq. (3.3).

3.6 Summary

A summary of the final approach from a compute graph point of view, is given in Figure 3.1. The proposed approach aims to realize the goal of learning to generate *good* training samples by evaluating the synthesis quality through the lens of a closed-form trainable probe model, the prediction loss of which is used as a loss for the G updates. Therefore, G is expected to be progressively guided *towards* producing

realistic, relevant and informative samples, through the reinforcement of which may vary depending on the inherent nature of the chosen probe model.

CHAPTER 4

EXPERIMENTS

We present a thorough experimental evaluation on GZSL benchmarks in which the results show that the proposed sample probing approach yields improvements when introduced into state-of-the-art baselines. In this chapter, we first explain the experimental setup in detail. Then, we present the results of our extensive experiments, comparisons, and analyses.

4.1 Experimental setup

The lack of a consensus on setting details for the experimental setup poses a serious threat to making accurate and consistent comparisons across studies in the field of (G)ZSL. Elaborately specifying training details such as training schemes and hyper-parameter tuning strategies is an aspect that should be emphasized to be able to reproduce experimental results as accurately as possible and to make meaningful comparisons across studies. In particular, the principled tuning of hyper-parameters, which naturally have a serious impact on the test result, is of high importance. Below, we present the details of our experimental setup including our proposed principled hyper-parameter tuning policy.

4.1.1 Datasets

We use four widely used GZSL benchmark datasets: Caltech-UCSD-Birds (CUB) [63], Oxford Flowers (FLO) [64], SUN Attribute (SUN) [65] and Animals with Attributes 2 (AWA2, more simply AWA) [8], for the experiments.

Table 4.1: Statistics for CUB, FLO, SUN and AWA in terms of size, granularity, number of images, number of attributes, and number of classes as in proposed splits defined by [8].

Dataset	Size	Granularity	# of images	# of attributes	# of classes		
					train + val	test	all
CUB [63]	medium	fine	11 788	312	100 + 50	50	200
FLO [64]	medium	fine	8 189	1024	62 + 20	20	102
SUN [65]	medium	fine	14 340	102	580 + 65	72	717
AWA [8]	medium	coarse	37 322	85	27 + 13	10	50

CUB [63] is a medium-scale fine-grained dataset consisting of 11 788 images of 200 classes of birds species such as *chipping sparrow*, *tropical kingbird*, *summer tanager*, etc. Attribute annotations of the CUB dataset have 312 dimensions, indicating *bill shape*, *eye color*, *size*, *breast pattern*, etc.

FLO [64] is a medium-scale fine-grained dataset consisting of 8 189 images of 102 classes of flowers such as *balloon flower*, *magnolia*, *water lily*, etc. Each class consists of between 40 and 258 images. Attribute annotations of the FLO dataset have 1024 dimensions.

SUN [65] is a medium-scale fine-grained dataset consisting of 14 340 images of 717 classes of scene types such as *airfield*, *waiting room*, *zoo*, etc. Attribute annotations of the SUN dataset have 102 dimensions, indicating *working*, *metal*, *open area*, *stressful*, etc.

AWA [8] is a medium-scale coarse-grained dataset consisting 37 322 images of 50 classes of animals such as *polar bear*, *tiger*, *zebra*, etc. Attribute annotations of the AWA dataset have 85 dimensions, indicating *brown*, *stripes*, *water*, *eats fish*, etc.

Following the state-of-the-art, we use the class embeddings and the proposed splits version 2.0 defined by [8]. Table 4.1 summarizes the detailed statistics of each aforementioned dataset.

In our experiments, we use the image features extracted from ResNet-101 backbone

pre-trained on ImageNet 1K. In the experiments based on *fine-tuned* representations, we use the backbone network fine-tuned with the training images of seen classes, as in [13, 18].

4.1.2 Evaluation metrics

In contrast to ZSL, GZSL evaluation includes not only the test classes but also the training classes. We evaluate the results in terms of GZSL-u (u), GZSL-s (s), and h-score (H) values [8]. GZSL-u indicates the top-1 performance on unseen/test classes, whereas GZSL-s indicates the top-1 performance on seen/training classes. The h-score, *i.e*. the harmonic mean of GZSL-u and GZSL-s scores, aims to measure how well a model recognizes seen and unseen classes collectively.

4.1.3 Sample probing hyper-parameters

All the hyper-parameters of baseline generative ZSL models are kept unchanged except for the number of training iterations. Apart from the number of training iterations, we only tune the hyper-parameters of the sample probing model, on the validation set. We tune (i) the number of meta-learning tasks, (ii) the number of different sets of probe-validation classes, (iii) the number of probe-train classes, (iv) the number of probe-validation classes, (v) the number of synthetic samples generated for probe-train classes, (vi) the number of probe-validation samples, (vii) the sample-probing loss weight, (viii) the sample-probing loss type, (ix) closed-form probe model type, (x) the regularization parameters of ESZSL [46] and (xi) the weighting coefficients of Vis2Sem [61] and Sem2Vis [62].

4.1.4 Hyper-parameter tuning policy

In our preliminary studies, we observe that the final GZSL performance, especially in terms of h-score, of most models, strongly depends on the selection of the hyper-parameters. We also observe that there is no widely-accepted policy on how the hyper-parameters of GZSL models shall be tuned. It is a rather common practice

in the GZSL literature to either directly report the hyper-parameters used in experiments without an explanation on the tuning strategy or simply refer to *tuning on the validation set*, which we find a vaguely-defined policy as (i) [8] defines an unseen-class only validation split, which does not allow monitoring the h-score, and (ii) it is unclear which metric one should use for GZSL model selection purposes.

The main factor that complicates model selection in GZSL is the fact that the degree of fitting to the training set can heavily affect the balance between making seen-class versus unseen-class predictions at test time and may significantly alter the resulting h-score values, even in the case of generative approaches. Therefore, for instance, measuring only ZSL accuracy on a validation set with unseen classes only may yield suboptimal results.

Therefore, to obtain comparable results within our experiments, we use the following policy to tune the hyper-parameters of our approach and our baselines: we first leave out 20% of train class samples as *val-seen* samples. We periodically train a supervised classifier by taking synthetic samples from the generative model and evaluating it on the validation set, consisting of the aforementioned *val-seen* samples plus the *val-unseen* samples with respect to the benchmark splits. We choose the hyper-parameter combination with the highest h-score on the validation set. We obtain final models by re-training the generative model from scratch on the training and validation examples combined using the selected hyper-parameters.

4.2 Main results

In this section, we discuss our main experimental results. As we observe that the results are heavily influenced by the hyper-parameter tuning strategy, our main goal throughout our experiments is the validation of the proposed sample probing idea by integrating it into strong generative GZSL baselines and then comparing results using the same tuning methodology. Using this principle, we present two main types of analysis: (i) the evaluation of the proposed approach using ESZSL as the probe model in combination with a number of generative GZSL models, and (ii) the evaluation of alternative closed-form probe models within our framework.

4.2.1 Generative GZSL models with sample probing

To evaluate the sample probing approach as a general technique to improve generative model training, we integrate it into four recent generative GZSL approaches: conditional Wasserstein GAN (cWGAN) [66, 34], LisGAN [20], TF-VAEGAN [18] and FREE [19]. We additionally report results for the variant of TF-VAEGAN with the fine-tuned representations (*TF-VAEGAN-FT*), as it is the only one among them with reported fine-tuning results. For the cWGAN, we follow the implementation details described in [14], and tune hyper-parameters using our policy. For LisGAN, TF-VAEGAN and FREE models, we use the official repositories shared by their respective authors. We use the version of TF-VAEGAN without *feedback loop* [18], for simplicity, as the model yields excellent performance with and without feedback loop. In all models (except cWGAN), we only re-tune the number of training iterations of the original models using our hyper-parameter tuning policy, to make the results comparable, as it is unclear how the original values were obtained. [1] We keep all remaining hyper-parameters unchanged to remain as close as possible to the original implementations.

The results over the four benchmark datasets are presented in Table 4.2. In terms of the h-scores, we observe improvements in 17 out of 19 cases, at varying degrees (up to 4.6 points). Only in two cases, we observe a slight degradation (maximum of 0.2 points) in performance. Overall, these improvements over already strong and state-of-the-art (or competitive) baselines validate the effectiveness of the proposed sample probing approach, suggesting that it is a valid method towards end-to-end learning of generative GZSL models directly optimized for synthetic train data generation purposes.

4.2.2 Sample probing with alternative closed-form models

We now evaluate our approach with different closed-form probe models, specifically ESZSL[46], Sem2Vis [62], and Vis2Sem [61], as described in Section 3.4. For these experiments, we use the TF-VAEGAN as the base generative model.

[1] We also tune LisGAN for AWA2 as the original paper reports AWA1 results instead.

Table 4.2: Evaluation of sample probing with multiple generative GZSL models on four benchmark datasets. Each row pair shows the effect of adding sample probing to a particular generative GZSL model, using ESZSL as the closed-form probe model. We use the same hyper-parameter optimization policy in all cases to make results comparable. We observe h-score improvements at varying degrees in 17 out of 19 model, feature & dataset variations.

	Sample probing	CUB			FLO			SUN			AWA		
		u	s	**H**	u	s	**H**	u	s	**H**	u	s	**H**
cWGAN [34]	N	45.1	53.1	48.7	50.7	74.3	60.3	41.6	37.3	39.3	-	-	-
	Y (ESZSL)	48.2	52.4	**50.2**	51.8	74.1	**61.0**	44.4	36.6	**40.1**	-	-	-
LisGAN [20]	N	40.9	60.5	48.8	53.1	81.7	64.4	41.5	36.6	38.9	44.2	77.0	56.1
	Y (ESZSL)	44.2	59.2	**50.6**	56.7	77.8	**65.6**	44.0	35.4	**39.2**	46.2	71.5	**56.2**
TF-VAEGAN [18]	N	53.9	58.4	56.0	59.4	78.3	67.5	42.9	39.3	41.0	54.4	75.2	63.2
	Y (ESZSL)	51.1	63.3	**56.6**	63.5	83.2	**72.1**	44.0	39.7	**41.7**	55.2	74.7	**63.5**
TF-VEAGAN-FT [18]	N	64.2	72.7	68.2	70.0	91.3	79.2	46.5	41.7	**44.0**	41.7	90.2	57.0
	Y (ESZSL)	63.1	76.1	**69.0**	70.2	91.7	**79.5**	47.8	40.6	43.9	45.6	87.6	**60.0**
FREE [19]	N	51.2	61.5	**55.9**	62.8	80.7	70.6	46.2	37.2	41.2	48.2	78.7	59.8
	Y (ESZSL)	51.6	60.4	55.7	65.6	82.2	**72.9**	48.2	36.5	**41.5**	51.3	78.0	**61.8**

The results with four configurations over four benchmark datasets are presented in Table 4.3. First of all, in terms of h-scores, we observe considerable performance variations across the probe models and datasets: Sem2Vis performs the best on CUB ($+0.9$ over the baseline), ESZSL provides a clear gain on FLO ($+4.6$) and a relative improvement on AWA ($+0.3$), and Vis2Sem improves the most on SUN ($+1.8$). These results suggest that sample probe alternatives have their advantages and disadvantages, and their performances can be data-dependent. Therefore, in a practical application, probe model options can be incorporated into the model selection process.

More in-depth understanding of closed-form model characteristics for sample probing purposes, and the formulation and evaluation of other probe models can be important future work directions. Overall, the fact that we observe equivalent (2) or better (9) h-scores in 11 out of 12 sample probing experiments indicates the *versatility* of the approach in terms of compatibility with various closed-form probe models.

Table 4.3: Sample probing with alternative closed-form models, based on TF-VAEGAN.

Closed-form	CUB			FLO			SUN			AWA		
probe model	u	s	**H**	**u**	s	**H**	**u**	s	**H**	**u**	s	**H**
-	53.9	58.4	56.0	59.4	78.3	67.5	42.9	39.3	41.0	54.4	75.2	63.2
ESZSL	51.1	63.3	56.6	63.5	83.2	**72.1**	44.0	39.7	41.7	55.2	74.7	**63.5**
Sem2Vis	51.9	63.0	**56.9**	58.6	80.9	68.0	44.7	38.4	41.3	54.9	74.6	63.2
Vis2Sem	37.1	70.4	48.6	58.3	80.1	67.5	46.0	40.1	**42.8**	55.3	74.3	63.4

4.2.3 Comparison to other generative GZSL approaches

Performance comparisons across independent experiment results can be misleading due to differences in formulation-agnostic implementation and model selection details. Nevertheless, we present an overall comparison to the (other) state-of-the-art generative GZSL results.

In Table 4.4, we compare our results with the state-of-the-art generative approaches for GZSL. During training, we select our best model based on the validation results and report test results on models that give the best validation scores. For consistency and to keep the baseline comparable to our results, we again report our results for LisGAN, TF-VAEGAN (without feedback loop), and FREE using our hyper-parameter tuning policy, but do acknowledge that the original papers typically report higher results. The upper part of the table contains results with the original image representations, and the lower part contains those based on fine-tuned representations.

From the results without fine-tuning, we observe that the proposed sample probing-based generative model yields state-of-the-art h-scores in all CUB, FLO, SUN and AWA datasets. We also observe competitive results in terms of individual unseen and seen class accuracy values. When compared against results using fine-tuned representations, we again observe state-of-the-art h-scores on CUB and FLO datasets, with a close second on SUN. On AWA, we observe that f-VAEGAN achieves the highest results with a significant margin over our TF-VAEGAN based baseline, where the sample probing improves the baseline yet still achieves a score below that of f-VAEGAN.

Table 4.4: Comparison against state-of-the-art generative model based GZSL on CUB, FLO, SUN and AWA datasets. Results obtained with the proposed features are reported, together with the results obtained with fine-tuned features under fine-tuned (FT). The results are reported in terms of top-1 accuracy of unseen (u) and seen (s) classes, together with their harmonic mean (H).

		CUB			FLO			SUN			AWA		
		u	s	**H**	u	s	**H**	u	s	**H**	u	s	**H**
	f-CLSWGAN [12]	3.7	57.7	49.7	59.0	73.8	65.6	42.6	36.6	39.4	57.9	61.4	59.6
	Cycle-WGAN [67]	47.9	59.3	53.0	61.6	69.2	65.2	**47.2**	33.8	39.4	**59.6**	63.4	59.8
	LisGAN [20]	40.9	60.5	48.8	53.1	81.7	64.4	41.5	36.6	38.9	44.2	77.0	56.1
	f-VAEGAN [13]	48.4	60.1	53.6	56.8	74.9	64.6	45.1	38.0	41.3	57.6	70.6	**63.5**
	TF-VAEGAN [18]	53.9	58.4	56.0	59.4	78.3	67.5	42.9	39.3	41.0	54.4	75.2	63.2
	Meta-VGAN [42]	**55.2**	48.0	53.2	-	-	-	-	-	-	57.4	70.5	**63.5**
	FREE [19]	51.2	61.5	55.9	62.8	80.7	70.6	46.2	37.2	41.2	48.2	78.7	59.8
	Ours (based on TF-VAEGAN)	51.1	**63.3**	**56.6**	**63.5**	83.2	**72.1**	44.0	**39.7**	**41.7**	55.2	74.7	**63.5**
	f-VAEGAN [13]	63.2	75.6	68.9	-	-	-	**50.1**	37.8	43.1	**57.1**	76.1	**65.2**
FT	TF-VAEGAN [18]	**64.2**	72.7	68.2	70.0	91.3	79.2	46.5	**41.7**	**44.0**	41.7	**90.2**	57.0
	Ours (based on TF-VAEGAN)	63.1	**76.1**	**69.0**	**70.2**	**91.7**	**79.5**	47.8	40.6	43.9	45.6	87.6	60.0

Overall, while it is hardly fair to compare models with significant implementation details, these results suggest the overall competitiveness of the obtained data generating models with sample probing.

4.3 Analysis

In the following, we present further analyses taking a more in-depth look at per-class seen and unseen performance gain of sample probing on the FLO dataset, discussing the effect of whether using ZSL or GZSL loss and observing the effect of sample probing loss weight on validation and test h-scores. Additionally, we also include quantitative and qualitative analyses of sample quality.

Table 4.5: Performance of baseline model with and without sample probing on FLO unseen classes (using TF-VEAGAN and ESZSL). Average per class top-1 accuracies of all unseen classes are listed.

Class ID	Baseline	Sample probing	Class ID	Baseline	Sample probing
001	25.0	25.0 (+0.0)	011	42.5	48.3 (+5.8)
002	23.3	36.7 (+13.4)	012	85.1	74.7 (-10.4)
003	55.0	50.0 (-5.0)	013	14.3	10.2 (-4.1)
004	55.4	53.6 (-1.8)	014	75.0	93.7 (+18.7)
005	86.2	87.7 (+1.5)	015	51.0	61.2 (+10.2)
006	75.6	77.8 (+2.2)	016	82.9	75.6 (-7.3)
007	72.5	77.5 (+5.0)	017	55.3	68.2 (+12.9)
008	92.9	96.5 (+3.6)	018	53.7	56.1 (+2.4)
009	37.0	43.5 (+6.5)	019	77.6	81.6 (+4.0)
010	75.6	86.7 (+11.1)	020	51.8	66.1 (+14.3)

4.3.1 Per class performance of sample probing for seen and unseen classes in FLO

Sample probing improves the GZSL results of TF-VAEGAN baseline on FLO with the performance gain in terms of GZSL-u ($+4.1$), GZSL-s ($+4.9$) and h-score ($+4.6$). While these results are informative about the overall performance of the approach, it can be insightful to observe the per class performance of the sample probing for seen and unseen classes since the h-score is a harsh metric.

In table 4.5, we can observe the performance of the baseline TF-VAGEAN model with and without sample probing for 20 unseen classes in the FLO dataset in terms of average top-1 accuracy. With the integration of the sample probing scheme, the performance of the baseline model increases in 14 out of 20 classes, while 6 of them have double-digit improvements ($+18.7$, $+14.3$, $+13.4$, $+12.9$, $+11.1$ and $+10.2$). On the other hand, the performance of the baseline model decreases in 5 classes (with the highest accuracy drop of -10.4) while remaining the same for a single class. Since the final classification models tend to yield higher confidence scores through seen classes in the GZSL setting, performance gains in the majority of the unseen classes show the superiority of the proposed sample probing.

We can make similar observations for 82 seen classes in FLO, listed in Table 4.6

Table 4.6: Performance of baseline model with and without sample probing on FLO seen classes (using TF-VEAGAN and ESZSL). Average per class top-1 accuracies of all seen classes are listed.

Class ID	Baseline	Sample probing	Class ID	Baseline	Sample probing
021	87.5	87.5 (+0.0)	062	36.4	72.7 (+36.3)
022	75.0	75.0 (+0.0)	063	100.0	100.0 (+0.0)
023	88.9	94.4 (+5.5)	064	90.0	100.0 (+10.0)
024	75.0	87.5 (+12.5)	065	90.0	95.0 (+5.0)
025	100.0	100.0 (+0.0)	066	100.0	100.0 (+0.0)
026	75.0	75.0 (+0.0)	067	62.5	75.0 (+12.5)
027	75.0	75.0 (+0.0)	068	36.4	54.5 (+18.1)
028	76.9	76.9 (+0.0)	069	90.9	100.0 (+9.1)
029	93.7	100.0 (+6.3)	070	100.0	100.0 (+0.0)
030	94.1	94.1 (+0.0)	071	87.5	87.5 (+0.0)
031	50.0	60.0 (+10.0)	072	52.6	57.9 (+5.3)
032	22.2	77.8 (+55.6)	073	89.7	94.9 (+5.2)
033	88.9	88.9 (+0.0)	074	73.5	67.6 (-5.9)
034	87.5	100.0 (+12.5)	075	91.7	100.0 (+8.3)
035	88.9	88.9 (+0.0)	076	76.2	81.0 (+4.9)
036	53.3	53.3 (+0.0)	077	96.0	96.0 (+0.0)
037	100.0	100.0 (+0.0)	078	93.0	93.0 (+0.0)
038	81.8	90.9 (+9.1)	079	100.0	100.0 (+0.0)
039	25.0	50.0 (+25.0)	080	90.5	90.5 (+0.0)
040	61.5	76.9 (+15.4)	081	87.9	90.9 (+3.0)
041	96.0	100.0 (+4.0)	082	63.6	72.7 (+9.1)
042	50.0	58.3 (+8.3)	083	80.8	88.5 (+7.7)
043	69.2	73.1 (+3.9)	084	47.1	70.6 (+23.5)
044	100.0	94.7 (-5.3)	085	46.2	53.8 (+7.6)
045	62.5	75.0 (+12.5)	086	66.7	75.0 (+8.3)
046	97.4	97.4 (+0.0)	087	76.9	84.6 (+7.7)
047	92.3	92.3 (+0.0)	088	83.9	90.3 (+6.4)
048	78.6	92.9 (+14.3)	089	91.9	94.6 (+2.7)
049	100.0	100.0 (+0.0)	090	68.7	75.0 (+6.3)
050	22.2	44.4 (+22.2)	091	53.3	66.7 (+13.4)
051	82.7	84.6 (+1.9)	092	76.9	76.9 (+0.0)
052	88.2	88.2 (+0.0)	093	77.8	77.8 (+0.0)
053	63.2	73.7 (+10.5)	094	96.9	96.9 (+0.0)
054	91.7	91.7 (+0.0)	095	88.5	84.6 (-3.9)
055	71.4	64.3 (-7.1)	096	44.4	44.4 (+0.0)
056	100.0	100.0 (+0.0)	097	61.5	61.5 (+0.0)
057	69.2	69.2 (+0.0)	098	81.2	62.5 (-18.7)
058	100.0	100.0 (+0.0)	099	84.6	76.9 (-7.7)
059	84.6	100.0 (+15.4)	100	90.0	90.0 (+0.0)
060	100.0	100.0 (+0.0)	101	91.7	91.7 (+0.0)
061	100.0	100.0 (+0.0)	102	80.0	80.0 (+0.0)

Table 4.7: ZSL vs GZSL based sample probing losses, (using TF-VAEGAN and ESZSL).

	Baseline			Sample probing					
				zsl-loss			gzsl-loss		
	u	*s*	**H**	u	*s*	**H**	u	s	**H**
CUB	53.9	58.4	56.0	50.5	63.6	56.3	51.1	63.3	**56.6**
FLO	59.4	78.3	67.5	62.4	83.8	71.5	63.5	83.2	**72.1**
SUN	42.9	39.3	41.0	44.0	39.7	**41.7**	46.0	36.9	41.0
AWA	54.4	75.2	63.2	55.2	74.7	**63.5**	55.6	72.8	63.0

in terms of average top-1 accuracy. The performance of the baseline TF-VAEGAN remains the same for 37 classes, most of them are already classified highly accurately. The baseline model with integrated sample probing improves the results for a lot of seen classes with a significant ratio of 39 out of 82. Performance gain for *Class-032* reaches a remarkable $+55.0$ bringing up the average top-1 accuracy from 22.2 to 77.8. 6 out of 82 seen classes experience performance decrease with sample probing where the highest drop is measured as -18.7 bringing down the accuracy of *Class-098* from 81.2 to 62.5.

4.3.2 ZSL vs GZSL loss in sample probing

We define two different types of losses (*zsl-loss* and *gzsl-loss*) used as L_{pb} in Eq.3.3. They differ from each other in terms of classes among which the real examples of probe-validation classes are classified, during the evaluation of the sample probing ZSL model. *zsl-loss* and *gzsl-loss* indicate that the examples of probe-validation classes are classified among only probe-validation classes, and both probe-train and probe-validation classes, respectively.

In Table 4.7, we present a comparison of our approach, using TF-VAEGAN as the generative model and ESZSL as the probe model when *zsl-loss* and *gzsl-loss* used as L_{pb}. We observe that using either one during the evaluation of the sample probing ZSL model, brings its own characteristic results. On all datasets, using *zsl-loss* in-

Table 4.8: Selected sample probing loss types for GZSL models using ESZSL as the closed-form probe model on CUB, FLO, SUN and AWA datasets.

	CUB	FLO	SUN	AWA
cWGAN [34]	gzsl	gzsl	zsl	-
LisGAN [20]	gzsl	gzsl	zsl	zsl
FREE [19]	gzsl	gzsl	zsl	zsl
TF-VAEGAN [18]	gzsl	gzsl	zsl	zsl
TF-VAEGAN-FT [18]	gzsl	zsl	zsl	gzsl

creases the seen accuracy while using *gzsl-loss* increases the unseen accuracy. We choose among these two options using our same hyper-parameter tuning policy, on the validation set.

Table 4.8 also reveals an interesting pattern showing that independent from the generative model being used, a single version of the loss tends to be selected in each dataset. The only exceptions are observed with the fine-tuned features, which is not unusual considering that the visual data is (almost) completely different. This pattern highlights that this choice is data-dependent, most likely due to various non-trivial factors.

4.3.3 Effect of sample probing loss weight

We observe the effects of sample probing loss weight on h-score in the validation and test set results to gain more insight into the challenging nature of model tuning in the ZSL setting.

Figure 4.1 shows the validation and test set h-score values as a function of sample probing loss weight. In the test set results, we observe an overall increasing performance trend with larger loss weights, up to the weight 6, highlighting the contribution of sample probing. The optimal weight with respect to the validation and the test sets, however, differs. In the validation set results, the maximum value of the h-score is obtained when the sample probing loss weight is set to 5. This observation is an example

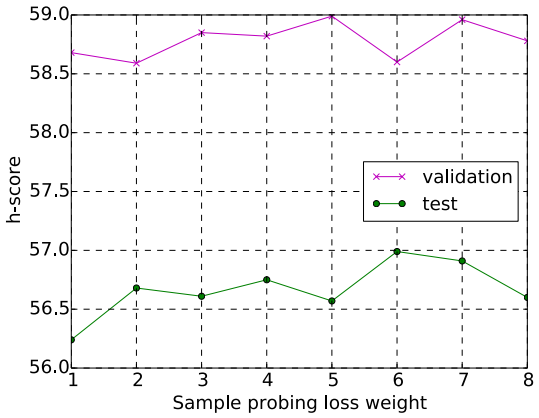

Figure 4.1: The effect of sample probing loss weight on the CUB dataset. Loss weight is set to 5 in test time concerning the best performance in validation time.

Table 4.9: Comparison of mean per-class Fréchet Distance between real and generated unseen class samples on CUB, AWA and FLO datasets for TF-VAEGAN and our approach. Lower is better.

	CUB	FLO	AWA
Baseline	21.5	31.0	18.5
Ours	**19.8**	**30.2**	**17.9**

of the difficulty of tuning the ZSL model based on the validation set. Still, we set the loss weight to 5 following our hyper-parameter tuning policy, which yields almost 0.5 lower than the maximum test-set score observed for this single hyper-parameter.

4.3.4 Quantitative analysis of sample quality

We quantitatively evaluate the sample quality using Fréchet (Wasserstein-2) distance, which is also used in the FID metric for evaluating GANs.

In Table 4.9, we provide a comparison for TF-VAEGAN and our approach for respective mean per-class Fréchet distances between real and synthetic samples (200 synthetic samples per class) of unseen classes on CUB, FLO and AWA datasets. Lower distance scores indicate better sample quality.

From the table, we can observe that the sample probing provides lower Fréchet distances over TF-VAEGAN baseline on all three CUB (-1.7), FLO (-0.8) and AWA (-0.6) datasets indicating that the sample quality of generator trained with sample probing is better. Overall, the results show that sample probing helps the generator to generate more realistic samples compared to TF-VAEGAN.

4.3.5 Qualitative analysis of sample quality

We further investigate the qualitative analysis of the sample quality to provide additional insight into the improvements that can be gained using the proposed sample probing scheme.

We present t-SNE visualizations of synthetic class samples in Figure 4.2, which can be useful in the presence of quantitative metrics such as the h-score and the Wasserstein-2 distance. t-SNE plots visualize what kind of improvements can be achieved in terms of the learned manifolds. In the figure, each plot corresponds to an unseen class on the FLO dataset (*pink primrose*, *canterbury bells*, *sweet pea*, *globe thistle*, *spear thistle* and *yellow iris*, respectively), and the points correspond to the t-SNE embeddings of real samples ($\times$ points), generated samples using TF-VAEGAN with sample probing (points) and those using the baseline TF-VAEGAN model without sample probing (▲ points).

From the plots, we can observe that the generative model trained with sample probing tends to yield samples much more aligned with the corresponding true class distributions, compared to those of the baseline model. Overall, these plots demonstrate how sample probing can improve the overall sample quality of a generative model, and possibly lead to superior recognition models when the generated samples are used for classifier training.

4.3.6 Training time

As the sample probing model involves using a closed-form solver at every single generative model update step, it does have an extra cost. For example, training TF-

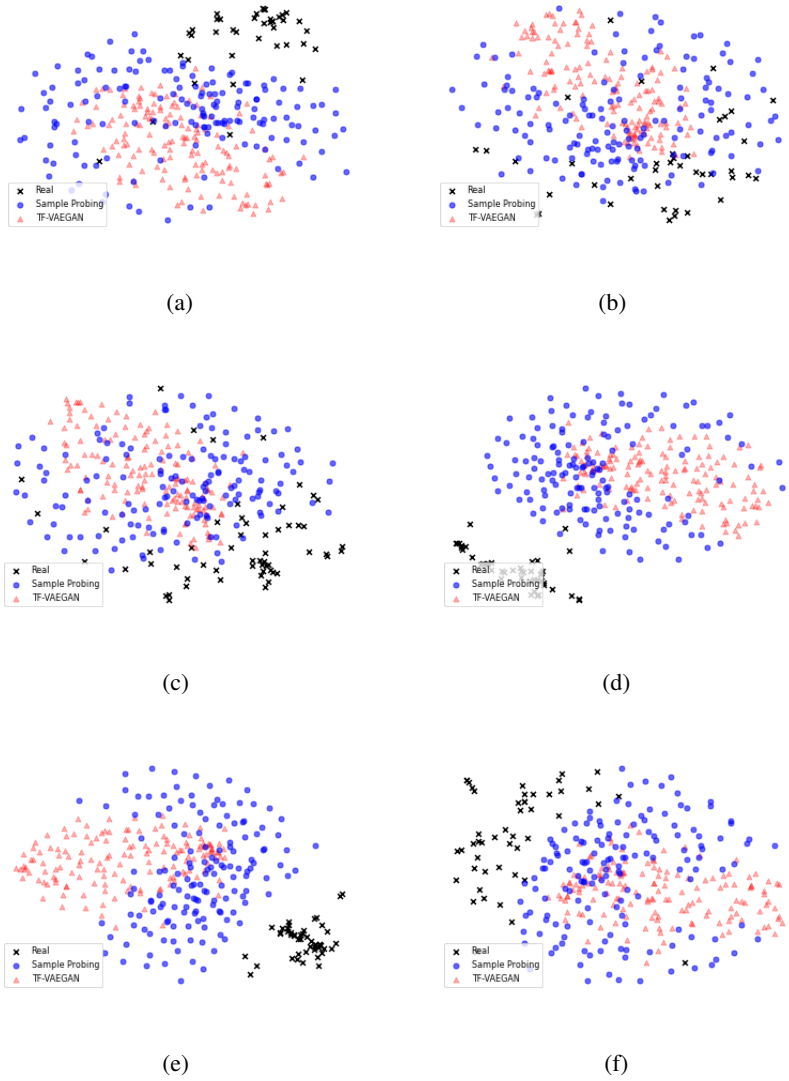

Figure 4.2: t-SNE visualization of different unseen classes from FLO dataset. Each plot shows t-SNE embeddings of real samples (× points), generated samples using TF-VAEGAN with sample probing (points) and those using the baseline TF-VAEGAN [18] model without sample probing (▲ points).

VAEGAN for 100 epochs on the CUB dataset takes 27.2 minutes without sample probing, and 43.2 minutes with sample probing. Despite its overhead, since all the experiments are in the feature domain as in other mainstream generative GZSL studies, the overall experiment durations remain practically feasible, in the order of 1 to 5 hours (depending on the maximum number of iterations, and the dataset size).

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this chapter, we conclude the studies that constitute the thesis and we discuss the ideas about potential future works and promising contributions that can be used to expand and make the proposed approach in this thesis more valuable.

5.1 Conclusion

We propose a principled GZSL approach, namely *sample probing*, which makes use of closed-form ZSL models in generative model training to provide a sample-driven and end-to-end feedback to the generator. Sample probing aims to directly maximize the value of training examples for ZSL training purposes and can easily be integrated into existing generative GZSL approaches.

In Chapter 3, we investigate the already existing competitive generative zero-shot learning model baselines into which the sample probing scheme is integrated, in detail and we formulate the proposed sample probing scheme as a simple yet powerful meta-learning approach and investigate several alternative closed-form solvers. We then show that the resulting compute graph is both efficient and end-to-end differentiable that generative model parameters can be updated with training signals produced by probing models with exact closed-form solutions.

The experiments over four benchmark datasets with four recent and competitive generative GZSL approaches show that the proposed sample probing scheme consistently improves the GZSL results. Additional quantitative and qualitative analyses also point out the increase in the overall sample quality. Extensive experiments show

that further performance gain can be achieved with various closed-form probe models that can be easily incorporated into the model selection process, indicating the versatility of the approach in terms of compatibility. According to performance comparisons across independent experiments, sample probing achieves state-of-the-art results when integrated into state-of-the-art baselines.

Additionally, we elaborate on the details of the hyper-parameter selection process in the generalized zero-shot learning setting. Through our studies in GZSL, we experience difficulties in hyper-parameter tuning policy and, naturally, comparisons across independent studies. Unfortunately, the proposed tuning policy is vaguely defined as discussed in Chapter 4 and, to the best of our knowledge, there is no comprehensive work focusing on this problem. We repeatedly observed that final results are heavily influenced by the hyper-parameter tuning strategy which makes it even more important to follow a principled tuning policy. In this thesis, we present a principled hyper-parameter tuning policy, defined in Chapter 4, that makes the comparisons and analyses made in this study more accurate. We believe that the proposed hyper-parameter tuning policy will be illuminating for future studies and will lead to consistent results in the field.

5.2 Future work

While we mainly focus on integrating sample probing scheme into existing generative GZSL approaches in this thesis, we already observe the unique characteristics of ESZSL [46], Vis2Sem [61] and Sem2Vis [62] on four benchmark GZSL datasets in Chapter 4. More in-depth analyses of closed-form model characteristics and the formulation and evaluation of other probe models such as expanding Table 4.3 to other generative models and introducing new closed-form solvers into sample probing scheme, can be important future work directions.

More detailed and comprehensive ablation studies focusing on the analyses of hyper-parameters of sample probing scheme such as the effects of the selection of sample probing loss type on the training of other closed-form solvers, the number of synthetic samples generated for closed-form solver training in each iteration, etc. can be worth

to explore in future.

Another potential future work direction can be the evaluation of the sample probing scheme under different configurations on zero-shot and other weakly supervised learning settings.

REFERENCES

[1] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning* (D. Precup and Y. W. Teh, eds.), vol. 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135, PMLR, 06–11 Aug 2017.

[2] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," 2017.

[3] Y. Tian, Y. Wang, D. Krishnan, J. B. Tenenbaum, and P. Isola, "Rethinking few-shot image classification: a good embedding is all you need?," 2020.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020.

[5] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," 2017.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2017.

[7] W.-L. Chao, S. Changpinyo, B. Gong, and F. Sha, "An Empirical Study and Analysis of Generalized Zero-Shot Learning for Object Recognition in the Wild," in *Proc. European Conf. on Computer Vision*, 2016.

[8] 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 Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2251–2265, 2018.

[9] J. Weston, S. Bengio, and N. Usunier, "Wsabie: Scaling up to large vocabulary image annotation," 2011.

[10] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, and T. Mikolov, "DeViSE: A Deep Visual-Semantic Embedding Model," in *Proc. Adv. Neural Inf. Process. Syst.*, pp. 2121–2129, 2013.

[11] A. Mishra, M. S. K. Reddy, A. Mittal, and H. A. Murthy, "A Generative Model For Zero Shot Learning Using Conditional Variational Autoencoders," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. Workshops*, Sept. 2017.

[12] Y. Xian, T. Lorenz, B. Schiele, and Z. Akata, "Feature generating networks for zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

[13] Y. Xian, S. Sharma, B. Schiele, and Z. Akata, "f-vaegan-d2: A feature generating framework for any-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 10275–10284, 2019.

[14] M. B. Sariyildiz and R. G. Cinbis, "Gradient matching generative networks for zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 2168–2178, 2019.

[15] Y. Zhu, M. Elhoseiny, B. Liu, X. Peng, and A. Elgammal, "A Generative Adversarial Approach for Zero-Shot Learning from Noisy Texts," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, (Salt Lake City, UT, USA), pp. 1004–1013, June 2018.

[16] E. Schonfeld, S. Ebrahimi, S. Sinha, T. Darrell, and Z. Akata, "Generalized Zero- and Few-Shot Learning via Aligned Variational Autoencoders," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, (Long Beach, CA, USA), pp. 8239–8247, June 2019.

[17] G. Arora, V. K. Verma, A. Mishra, and P. Rai, "Generalized zero-shot learning via synthesized examples," *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 4281–4289, 2018.

[18] S. Narayan, A. Gupta, F. S. Khan, C. G. M. Snoek, and L. Shao, "Latent Embedding Feedback and Discriminative Features for Zero-Shot Classification," in *Proc. European Conf. on Computer Vision*, pp. 479–495, 2020.

[19] S. Chen, W. Wang, B. Xia, Q. Peng, X. You, F. Zheng, and L. Shao, "FREE: Feature refinement for generalized zero-shot learning," in *Proc. IEEE Int. Conf. on Computer Vision*, pp. 122–131, 2021.

[20] J. Li, M. Jing, K. Lu, Z. Ding, L. Zhu, and Z. Huang, "Leveraging the invariant side of generative zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 7402–7411, 2019.

[21] Y. Belinkov, N. Durrani, F. Dalvi, H. Sajjad, and J. Glass, "What do neural machine translation models learn about morphology?," *arXiv preprint arXiv:1704.03471*, 2017.

[22] M. Peters, M. Neumann, L. Zettlemoyer, and W.-t. Yih, "Dissecting contextual word embeddings: Architecture and representation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Oct.-Nov. 2018.

[23] J. Hewitt and P. Liang, "Designing and Interpreting Probes with Control Tasks," in *Proc. of the Empiricial Methods in Natural Language Processing*, Sept. 2019.

[24] Y. Xian, B. Schiele, and Z. Akata, "Zero-Shot Learning - The Good, the Bad and the Ugly," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Mar. 2017.

[25] H. Jiang, R. Wang, S. Shan, and X. Chen, "Transferable Contrastive Network for Generalized Zero-Shot Learning," in *Proc. IEEE Int. Conf. on Computer Vision*, Aug. 2019.

[26] S. Liu, M. Long, J. Wang, and M. I. Jordan, "Generalized Zero-Shot Learning with Deep Calibration Network," in *Proc. Adv. Neural Inf. Process. Syst.*, pp. 2005–2015, 2018.

[27] Y.-Y. Chou, H.-T. Lin, and T.-L. Liu, "Adaptive and Generative Zero-Shot Learning," in *Proc. Int. Conf. Learn. Represent.*, p. 14, 2021.

[28] V. Vapnik and V. Vapnik, *Statistical Learning Theory 156–160*. Wiley, New York, 1998.

[29] M. Elhoseiny and M. Elfeki, "Creativity Inspired Zero-Shot Learning," in *Proc. IEEE Int. Conf. on Computer Vision*, Apr. 2019.

[30] D. P. Kingma and M. Welling, "Stochastic gradient vb and the variational auto-encoder," in *Proc. Int. Conf. Learn. Represent.*, vol. 19, 2014.

[31] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proc. Int. Conf. Mach. Learn.*, pp. 1278–1286, PMLR, 2014.

[32] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[33] A. R. Zamir, T.-L. Wu, L. Sun, W. B. Shen, B. E. Shi, J. Malik, and S. Savarese, "Feedback networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 1308–1317, 2017.

[34] T. Miyato and M. Koyama, "cGANs with Projection Discriminator," in *Proc. Int. Conf. Learn. Represent.*, Feb. 2018.

[35] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[36] S. Gidaris and N. Komodakis, "Dynamic few-shot visual learning without forgetting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 4367–4375, 2018.

[37] H.-J. Ye, H. Hu, D.-C. Zhan, and F. Sha, "Few-shot learning via embedding adaptation with set-to-set functions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 8808–8817, 2020.

[38] J. Bronskill, J. Gordon, J. Requeima, S. Nowozin, and R. E. Turner, "Tasknorm: Rethinking batch normalization for meta-learning," in *Proc. Int. Conf. Mach. Learn.*, vol. 119 of *Proceedings of Machine Learning Research*, pp. 1153–1164, PMLR, 2020.

[39] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell, "Meta-learning with latent embedding optimization," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[40] A. Nichol, J. Achiam, and J. Schulman, "On First-Order Meta-Learning Algorithms," *arXiv e-prints*, p. arXiv:1803.02999, Mar. 2018.

[41] V. K. Verma, D. Brahma, and P. Rai, "Meta-Learning for Generalized Zero-Shot Learning," *AAAI Conference on Artificial Intelligence*, vol. 34, pp. 6062–6069, Apr. 2020.

[42] V. K. Verma, A. Mishra, A. Pandey, H. A. Murthy, and P. Rai, "Towards Zero-Shot Learning With Fewer Seen Class Examples," in *WACV*, p. 11, 2021.

[43] Y. Yu, Z. Ji, J. Han, and Z. Zhang, "Episode-Based Prototype Generating Network for Zero-Shot Learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, (Seattle, WA, USA), pp. 14032–14041, June 2020.

[44] Y.-Y. Chou, H.-T. Lin, and T.-L. Liu, "Adaptive and generative zero-shot learning," in *Proc. Int. Conf. Learn. Represent.*, 2021.

[45] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. Int. Conf. Learn. Represent.*, 2018.

[46] B. Romera-Paredes and P. H. Torr, "An embarrassingly simple approach to zero-shot learning," in *Proc. Int. Conf. Mach. Learn.*, pp. 2152–2161, 2015.

[47] L. Bertinetto, P. H. S. Torr, J. Henriques, and A. Vedaldi, "Meta-Learning with Differentiable Closed-Form Solvers," in *ICLR*, p. 15, 2019.

[48] G. Bhat, F. J. Lawin, M. Danelljan, A. Robinson, M. Felsberg, L. Van Gool, and R. Timofte, "Learning What to Learn for Video Object Segmentation," in *Proc. European Conf. on Computer Vision*, May 2020.

[49] Y. Liu, L. Liu, H. Zhang, H. Rezatofighi, Q. Yan, and I. Reid, "Meta learning with differentiable closed-form solver for fast video object segmentation," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 8439–8446, IEEE, 2020.

[50] B. Hariharan and R. Girshick, "Low-shot Visual Recognition by Shrinking and Hallucinating Features," *arXiv e-prints*, p. arXiv:1606.02819, June 2016.

[51] Y.-X. Wang, R. Girshick, M. Hebert, and B. Hariharan, "Low-Shot Learning from Imaginary Data," *arXiv:1801.05401 [cs]*, Jan. 2018.

[52] 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 31*, pp. 983–993, 2018.

[53] E. Schwartz, L. Karlinsky, J. Shtok, S. Harary, M. Marder, A. Kumar, R. Feris, R. Giryes, and A. Bronstein, "Delta-encoder: an effective sample synthesis method for few-shot object recognition," in *NeurIPS*, pp. 2845–2855, 2018.

[54] M. Lazarou, Y. Avrithis, and T. Stathaki, "Few-shot learning via tensor hallucination," in *ICLR2021 workshop: "Synthetic Data Generation: Quality, Privacy, Bias"*, Apr. 2021.

[55] R. ZHANG, T. Che, Z. Ghahramani, Y. Bengio, and Y. Song, "Metagan: An adversarial approach to few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[56] K. Li and J. Malik, "Learning to optimize," *arXiv preprint arXiv:1606.01885*, 2016.

[57] Y. Chen, M. W. Hoffman, S. G. Colmenarejo, M. Denil, T. P. Lillicrap, M. Botvinick, and N. Freitas, "Learning to learn without gradient descent by gradient descent," in *Proc. Int. Conf. Mach. Learn.*, pp. 748–756, 2017.

[58] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, "Large-Scale Long-Tailed Recognition in an Open World," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, (Long Beach, CA, USA), pp. 2532–2541, June 2019.

[59] J. Ren, C. Yu, S. Sheng, X. Ma, H. Zhao, S. Yi, and H. Li, "Balanced Meta-Softmax for Long-Tailed Visual Recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, Nov. 2020.

[60] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, "Evaluation of Output Embeddings for Fine-Grained Image Classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 2927–2936, 2015.

[61] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4447–4456, 2017.

[62] Y. Shigeto, I. Suzuki, K. Hara, M. Shimbo, and Y. Matsumoto, "Ridge regression, hubness, and zero-shot learning," in *ECML/PKDD*, 2015.

[63] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," Tech. Rep. CNS-TR-2011-001, California Institute of Technology, 2011.

[64] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.

[65] G. Patterson and J. Hays, "Sun attribute database: Discovering, annotating, and recognizing scene attributes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 2751–2758, 2012.

[66] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, pp. 214–223, PMLR, 2017.

[67] R. Felix, B. G. V. Kumar, I. Reid, and G. Carneiro, "Multi-modal Cycle-consistent Generalized Zero-Shot Learning," in *Proc. European Conf. on Computer Vision*, July 2018.

TEZ İZİN FORMU / THESIS PERMISSION FORM

ENSTİTÜ / INSTITUTE

Fen Bilimleri Enstitüsü / Graduate School of Natural and Applied Sciences ☐

Sosyal Bilimler Enstitüsü / Graduate School of Social Sciences ☐

Uygulamalı Matematik Enstitüsü / Graduate School of Applied Mathematics ☐

Enformatik Enstitüsü / Graduate School of Informatics ☐

Deniz Bilimleri Enstitüsü / Graduate School of Marine Sciences ☐

YAZARIN / AUTHOR

Soyadı / Surname : .....
Adı / Name : .....
Bölümü / Department : .....

TEZİN ADI / TITLE OF THE THESIS (**İngilizce** / English) : .....
.....
.....
.....
.....

TEZİN TÜRÜ / DEGREE: **Yüksek Lisans** / Master ☐ **Doktora** / PhD ☐

1. **Tezin tamamı dünya çapında erişime açılacaktır. /** Release the entire work immediately for access worldwide. ☐
2. **Tez iki yıl süreyle erişime kapalı olacaktır.** / Secure the entire work for patent and/or proprietary purposes for a period of **two year. *** ☐
3. **Tez altı ay süreyle erişime kapalı olacaktır.** / Secure the entire work for period of **six months**. ***** ☐

***** *Enstitü Yönetim Kurulu Kararının basılı kopyası tezle birlikte kütüphaneye teslim edilecektir. A copy of the Decision of the Institute Administrative Committee will be delivered to the library together with the printed thesis.*

Yazarın imzası / Signature ..... **Tarih** / Date .....