Field-Guide-Inspired Zero-Shot Learning

Utkarsh Mall, Bharath Hariharan, Kavita Bala
Cornell University
{utkarshm, bharathh, kb}@cs.cornell.edu

Abstract

Modern recognition systems require large amounts of supervision to achieve accuracy. Adapting to new domains requires significant data from experts, which is onerous and can become too expensive. Zero-shot learning requires an annotated set of attributes for a novel category. Annotating the full set of attributes for a novel category proves to be a tedious and expensive task in deployment. This is especially the case when the recognition domain is an expert domain. We introduce a new **field-guide-inspired** *approach to zero-shot annotation where the learner model interactively asks for the most useful attributes that define a class. We evaluate our method on classification benchmarks with attribute annotations like CUB, SUN, and AWA2 and show that our model achieves the performance of a model with full annotations at the cost of significantly fewer number of annotations. Since the time of experts is precious, decreasing annotation cost can be very valuable for real-world deployment.*

1. Introduction

Modern recognition systems require vast amounts of labeled data. This is infeasible to acquire in many domains, especially when the classes in question involve subtle distinctions that require an expert: experts have limited time and availability and cannot annotate thousands of images. This has motivated research into *zero-shot learning* (ZSL) where the goal is to build effective recognition models from *class descriptions* alone. The name "zero-shot" suggests that the labeling effort for the annotator is zero. However, this is not true: in current ZSL systems, the annotator must specify for each class *hundreds* of attributes for *thousands* of classes (Figure 1 (left)). For example, in one of our preliminary experiments, it took an ornithologist more than **15 minutes** to fully describe the 312 different attributes in CUB; annotating all 200 classes would have taken the expert more than **50 hours**! For ZSL to truly decrease the annotator's burden, the cost of attribute description has to be significantly cheaper.

back-color back-pattern belly-color
belly-pattern bill-color bill-length
bill-shape breast-color
breast-pattern crown-color
eye-color forehead-color
head-pattern leg-color nape-color
primary-color shape size
tail-pattern tail-shape throat-color
under-tail-color underparts-color
upper-tail-color upperparts-color
wing-color wing-pattern

English Name: **Mistle Thrush**
Scientific Name: *Turdus viscivorus*
Description:
1. Plain greyish brown backs and neatly round-spotted underparts
2. Larger than the Song Thrush, but the breast has much less buff

Figure 1. **Left:** For each new class in the CUB dataset, an annotator must label 28 attributes (and 312 different quantities). The attributes are multidimensional and continuous-valued, **Right:** A *field guide* example of a bird class. There is no full attribute description; instead the bird is described by a few key attributes that distinguish it from close cousins.

Past work on addressing this concern has looked at using freely available text from the internet, e.g., Wikipedia[7, 16, 2, 23, 43], or relying on word embeddings of class names[3, 8, 24, 1]. While such "unsupervised" ZSL is intriguing, neither class names nor Wikipedia descriptions are intended to be used for *identifying* classes. Class names are terse and may often be based on location rather than appearance (e.g.,"Northern Royal Albatross"). Wikipedia articles might likewise include irrelevant information (e.g., about etymology) and miss vital visual details. As such, we find that these approaches result in a substantial reduction in accuracy of almost 20 points(see Figure 4 (middle row)). This suggests that expert-provided visual characteristics or attributes are indeed crucial for performance.

How can we record expert knowledge of class distinctions as completely as possible while still reducing their burden? One could simply reduce the *vocabulary* of attributes that the expert has to specify. But doing so might preclude important class distinctions, resulting in a dramatic reduction in accuracy. A better approach is provided by *field guides* that experts write to train *human* novices. In these guides, for each class, the expert first identifies a very similar previously defined class, and then specifies *only the most important attributes* that distinguish the two classes (Figure 1 (right)). This allows for concise descriptions that are both easy to write for the expert and also complete enough for the novice. What if one used these kinds of descriptions for ZSL?

While intuitive, in our experiments we find that this field guide-based approach is even worse than simply choosing a *random* smaller, fixed vocabulary of attributes to begin with. This is because it assumes that what the expert *thinks* are important attributes, are in fact what the machine finds relevant to make class distinctions. But machines are *not people*; the attributes that humans find most salient may not in fact be salient to machines. More broadly, what people find to be obvious class distinctions might appear extremely subtle to the machine and vice versa. Thus, we need a new approach that similarly reduces annotator effort by focusing only on important attributes, but that crucially relies on the machine to define attribute importance.

With these considerations in mind, we propose a new *learning interface* for ZSL learners to learn from experts. For each novel class, the expert first identifies a close cousin that the learner already knows about. The learner then chooses the attributes that it thinks will be most informative for it to learn and actively queries the expert for just these attributes. This learning space brings to the fore the question of query strategies: how must the learner choose attributes to query that maximize performance (achieve good accuracy) but minimize expert effort (choose informative attributes)? Similar problems are explored in active learning, where learners must choose unlabeled data points to label. However, where active learning techniques have a priori access to unlabeled examples to make a range of measurements (e.g., prediction uncertainty), in our case the learner must choose attributes to query for a *completely unseen class*.

We address this challenge by proposing multiple novel kinds of query strategies to learn from this new active ZSL interface. We design strategies based on a new measure of attribute uncertainty based on class taxonomies and a notion of expected change in model predictions. We also design strategies for when we have access to an *image* of the novel class, thus generalizing to the regime of combined zero- and few-shot learning. Our proposed query strategies can work out-of-the-box with existing zero-shot learners.

We experiment with three datasets, namely CUB, SUN and AWA2, and show significant reduction in annotation costs while keeping the performance on par with the full annotation models. With only 35% of annotations, we get close to full model performance on SUN and CUB. Our approach also significantly outperforms prior unsupervised ZSL work on CUB and SUN *without a single attribute annotation*. Our contributions are:

- A new *field-guide-inspired active ZSL* approach to collect expert annotations that is more *accurate* than using class names/textual descriptions and more *time-efficient* than annotating a full attribute description.
- New query strategies (e.g., based on uncertainty and expected model change), to actively query expert attributes to rapidly train the learner. Our results suggest that thinking about what information to acquire from the annotator, and what interface the expert uses, is a promising direction for building accurate recognition models that are easy to train.

2. Related Work

Zero-shot learning. In zero-shot learning [13, 14], the model learns to classify unseen classes without any training images by leveraging side information like attribute descriptions. Initial work by Lampert *et al*. [13] proposed first predicting attributes from images and then classifying images based on the predicted attributes. Recent work has looked at projecting image features into attribute space, and measuring similarity with class descriptions [8, 1, 25, 12, 31]. More recent work has used the attribute description to produce improvements by embedding these descriptions as well as images into a shared feature space [39, 6, 33, 20]. The auxilliary losses such as a reconstruction loss can be used to regularize the problem [31, 39]. Our proposed framework builds on these ideas, but leverages sparser but richer information about each class from the expert.

Because collecting attributes from experts is expensive, researchers have looked at other sources of class information. These methods have been colloquially referred to as “Unsupervised ZSL” as they do not require attribute supervision for test classes. They range from using word embeddings of the class names [8, 24, 1], class and attribute embedding [3], or textual descriptions (usually from Wikipedia) [7, 16, 2, 23, 43] of classes instead of attribute descriptions. Note that while these methods are called unsupervised, they still require information such as a large training corpus for word embedding or text articles (written by domain experts). In addition, these sources of information are not typically designed for *identifying* classes visually. While we share the same goal of reducing annotation cost, we propose an alternative active-learning framework for attribute annotation.

Our paper also considers the possibility of going beyond zero-shot, and assuming that the annotator can provide us with one image of the novel class. This notion of combining images with zero-shot attribute information was first introduced by Tsai *et al*. [32]. Schonfeld *et al*. [27] introduce a VAE model to align the image features and attributes in a latent space to perform combined few-shot and zero-shot learning. Since a field-guide has images as well as distinctive attributes, models combining these two are worth exploring.

Active learning. Active learning has been extensively explored in the the area of machine learning and computer vision. These methods aim to make judicial use of an annotation budget by selecting useful unlabeled data to be labelled first. Several methods have been proposed that pick data to be labelled based on objectives such as “bringing larger expected model change” [29, 10], “increasing the diversity of labelled set”[28, 40], “reducing the uncertainty in predictions” [17, 26, 9]. Recently many new techniques utilize adversarial learning to acquire labels for most informative data [41, 30, 42, 37]. Our work is inspired by some of these techniques, but applies to a completely different and new problem: that of choosing *attributes* to label. Since our

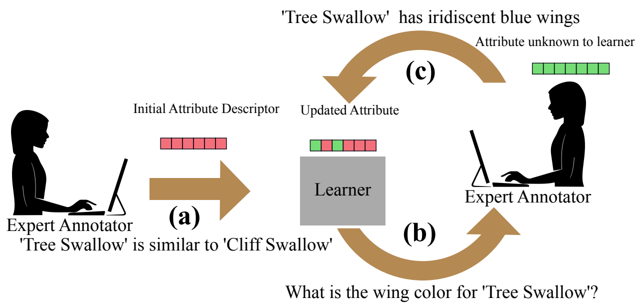
'Tree Swallow' has iridiscent blue wings
Attribute unknown to learner
(c)
Initial Attribute Descriptor
Updated Attribute
Learner
Expert Annotator
Expert Annotator
(a)
'Tree Swallow' is similar to 'Cliff Swallow'
(b)
What is the wing color for 'Tree Swallow'?

Figure 2. Overview of our *field-guide-inspired* workflow. (a) An expert annotator introduces the learner to a novel class by first providing a similar base class. (b) The Learner then interactively asks for values of different attributes using an Acquistion function. (c) The expert annotator provides the values of the attributes and the learner updates its state and class description.

methods do not have access to unlabelled data, the traditional active learning query strategies no longer apply. Nevertheless, we take inspiration from this prior work to define a family of new techniques more suited to the zero-shot setup.

Learning from experts. There are other pipelines that are motivated by the question of learning from experts. Misra et al. [18] propose a visual question answering system that learns by interactions. Unfortunately, it is not clear if this system can generalize to the general recognition setting. Earlier work leveraged experts to automatically correct part locations [4] or define patches for fine-grained classification [22]. However, these models rely explicitly on the part-based design of the recognition models. There has also been work on recognition techniques that query humans during *inference* for similarity comparisons [36] and attribute descriptions [34, 5]. In contrast, we query the human annotator for particular attributes when we are learning a novel class.

3. Method

3.1. Problem Setup

Our goal is to produce a learner that can learn new classes from very few interactions with an expert annotator. As in traditional zero-shot learning, we assume that the learner is first trained on some set of "base" or "seen" classes $\mathcal{B}$. For each base class $y \in \mathcal{B}$, the learner knows a vector of attributes $A(y) \in \mathbb{R}^d$, where d is the number of attributes (we assume real-valued attributes in this work, in line with previous work on ZSL). The learner is also provided with a large labeled *training set* for the base classes consisting of images $\mathbf{x}_i, i = 1, \ldots, n$ and corresponding labels y_i.

Once trained and deployed, the learner gets a set of hitherto unseen classes, $\mathcal{N}$ to recognize. It is here that our problem setup diverges from traditional zero-shot learning. A traditional zero-shot learning system needs the full attribute description of all novel classes: $\{A(y) \forall y \in \mathcal{N}\}$. In contrast, in our proposed setup, the learner must use *very few field-guide*-like interactions with the expert annotator.

In particular, for each new class $y \in \mathcal{N}$, the annotator first gives the learner the most similar class $S(y) \in \mathcal{B}$ from the set of base classes. Next, the learner incrementally asks the annotator for the values of attributes, one attribute at a time. The goal of the learner is then to learn to recognize the novel class from as few attribute queries as possible. We call this new kind of learning interface and the associated technique ZSL-Interactive(Figure 2).

To design a learner in this interactive setup, two questions must be answered: (a) how should we learn with the incomplete attribute description?, and (b) how should we choose which attributes to query? We address the first question in the next section and then discuss our strategy for the more challenging question of choosing attributes to annotate.

3.2. Learning from Sparse Attribute Annotation

For every novel class y, the learner is told the most similar class $S(y)$, and it queries values for a subset of attribute indices $I(y) \subseteq \{0 \cdots d\}$. For attributes where novel class information is missing, we impute the attribute descriptor using the values from $S(y)$.

$$A'(y)[i] = \begin{cases} A(y)[i] & i \in I(y) \\ A(S(y))[i] & i \notin I(y) \end{cases} \tag{1}$$

The imputed vector $A'(y)$ is used as the attribute descriptor for the novel class y in the zero-shot model. Note that this approach is model-agnostic and can be used with any zero-shot model out-of-the-box.

3.3. Strategies for Querying Attributes

We now describe how we pick the attributes iteratively to collect the sparse attribute annotation. Suppose the learner is learning about novel class y, with $S(y)$ as its most similar base class. Suppose it has already queried for a subset $I(y)$ of the attributes resulting in an imputed attribute vector $A'(y)$. Given this information, it must now choose a new attribute to query for. The learner will make this choice using a *query strategy* or *acquisition function* π; the attribute chosen is $\pi(S(y), I(y), A'(y))$.

The notion of a query strategy is reminiscent of active learning, where the learner must choose unlabeled data points that it wants labeled. However, in active learning, the learner has access to the unlabeled data, and so it can use its own belief about the unlabeled data points to make the call. For example, it can choose to label data points for which it is most uncertain, or for which an annotation is most likely to change its state significantly. In contrast, in our case, the learner is faced with a *completely unseen class*. How can the learner identify informative attributes when it has never seen this class before?

Below, we present two solutions to this challenge. The first solution uses a taxonomy over the base classes to find

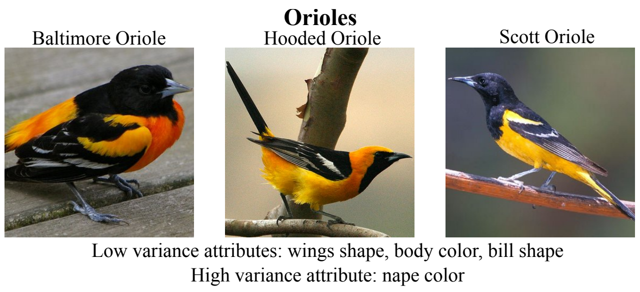
Orioles
Baltimore Oriole
Hooded Oriole
Scott Oriole
Low variance attributes: wings shape, body color, bill shape
High variance attribute: nape color

Figure 3. While attributes like "wing shape", "bill shape" are common to all orioles, color of the nape has high variance.

informative attributes. The second uses the imputed attribute vector and looks at the change in representation space with respect to the change in a attribute.

3.3.1 Taxonomy-Based Querying

Class taxonomies are common in many domains (e.g., birds) and can be defined easily by a domain expert. *Siblings* in this taxonomy are likely to be similar to each other, sharing several common attributes, and probably differing from each other in only a few attributes. For example, in the domain of birds, all classes of the "Oriole" supercategory have black wings, a yellow body, and conical bills. Therefore, it is not prudent to query for wing color, body color or bill shape for an Oriole class. Instead we might want to query for the nape color, which varies among Orioles: "Hooded Orioles" have a yellower nape than "Baltimore Orioles". Thus, in general, we want to query attributes that vary a lot among classses in the relevant subtree, and ignore those that don't.

We can use this intuition to intelligently pick attributes to query by leveraging the class taxonomy. Let R_z be the set of sibling classes for a base class z including the class z itself. When the annotator provides the similar base class $S(y)$ for the novel class y, we look at the *variance* of each attribute in sibling classes $R_{S(y)}$. Attributes with lower variance are common to all the sibling classes and will be less informative for a similar novel class y. In contrast, attributes with higher variance vary a lot in this subtree, so querying for their value is prudent. Thus, the *variance* for each attribute among the sibling categories $R_{S(y)}$ is a measure of which attributes are more informative for this set of classes.

We define this measure of variance amongst siblings as *Sibling-variance* for novel class y (denoted by $Q_{sv}(y)$):

$$Q_{sv}(y)[j] = \text{Var}(\{A(s_i)[j]; s_i \in R_{S(y)}\}) \tag{2}$$

where j indexes the attribute, Var denotes the variance of a set of values, and s_i ranges over the siblings of $S(y)$

The most informative attributes are the ones with maximum *Sibling-variance*. We therefore choose attributes to label in *decreasing order of Sibling-variance*:

$$\pi_{sv}(S(y), I(y), A'(y)) = \arg\max_{j \notin I(y)} Q_{sv}(y)[j] \tag{3}$$

Attributes sometimes belong to groups. For example, in CUB, many different attributes all correspond to body color: each attribute captures a different color, and together the body color attributes define a multinomial distribution over the different colors. For such groups of attributes, we only look at the maximum varying attribute to find *Sibling-variance*. Thus, if g is such a group of attributes, *Sibling-variance* for this group is be defined as,

$$Q_{sv}(y)[g] = \max_{j \in g} \text{Var}(\{A(s_i)[j]; s_i \in R_{S(y)}\}) \tag{4}$$

We select the whole group of attributes with maximum *Sibling-variance* for annotation.

Note that this method resembles uncertainty measure-based active learning methods. The learner first picks attributes for annotation where the learner is uncertain. But the way we measure this uncertainty is different from these methods, as we have no information about the novel class.

3.3.2 Querying Based on Representation Change

Many zero-shot learners encode both the attributes and images in a common latent space, and train classifiers in this space. Attribute encoders in these approaches can learn to perform a variety of useful functions, such as weighing attributes more if they are more identifiable or more discriminative.

Because the classifier operates in this latent space, annotations that significantly change this latent representation are more likely to influence classification decisions. This suggests that we should query attributes which when changed cause the largest change in latent space. We call this method of looking at changes in representation "*Representation-change*" denoted by Q_{rc}.

Concretely, let $\mathbf{E_a} : \mathbb{R}^d \to \mathbb{R}^l$, be the attribute encoder function that maps an attribute into a latent representation space of dimension l. Suppose that $A'(y)$ is the current imputed attribute representation (note that $A'(y)$ starts as $A(S(y))$ and is gradually filled in with the true attribute values $A(y)$ as the learner queries the annotator). This attribute vector is represented in latent space as $\mathbf{E_a}(A'(y))$. Now we wish to know which attributes when perturbed will lead to the largest change in this representation. Since the encoder architectures are non-linear, we make use of the local partial derivatives as represented by the Jacobian of the encoder to measure these changes.

$$\mathbf{J}_{\mathbf{E_a}}(\mathbf{x}) = \left[\frac{\partial \mathbf{E_a}(\mathbf{x})}{\partial \mathbf{x}_1} \cdots \frac{\partial \mathbf{E_a}(\mathbf{x})}{\partial \mathbf{x}_d} \right] \tag{5}$$

We define the *Representation-change* of attribute i as:

$$Q_{rc}(y|A'(y))[i] = \left\| \left. \frac{\partial \mathbf{E_a}(\mathbf{x})}{\partial \mathbf{x}_i} \right|_{\mathbf{x}=A'(y)} \right\|_2 \tag{6}$$

The l2-norm of the partial derivative measures how much a small perturbation to that particular attribute changes the encoding in the latent space. The learner then queries the attribute with the maximum *Representation-change*.

$$\pi_{rc}(S(y), I(y), A'(y)) = \arg\max_{j \notin I(y)} Q_{rc}(y|A'(y))[j] \quad (7)$$

For grouped attributes, similar to *Sibling-variance*, we measure *Representation-change* to be the maximum *Representation-change* within the attributes of that group.

Representation-change is similar to the "expected model change"-based active learning methods[29, 10]. These methods pick entities based on the most change to the model in expectation. Similarly, *Representation-change* picks attributes that change the encoding the most in latent space.

3.3.3 Querying Using Image Data

In practice, an expert can easily provide a single image of the novel class. Only recently have ZSL techniques begun to leverage this information[32, 27]. Here, we show how we can use this single image to better choose attributes to label, using the following *Image-based* strategy.subsection

The key intuition here is that the image provides a representation for the class that must ultimately match the final attribute description. As such, the learner can *recognize* attributes in the image, and attempt to reconcile differences between the recognized attributes and the imputed attribute description based on the expert annotation.

Concretely, suppose the image available for the novel class y is x_y. Suppose also that we have a trained *image encoder* $\mathbf{E_i}$ that maps the image to the latent space, and an *attribute decoder* $\mathbf{D_a}$ that decodes this latent representation into an attribute vector; many recent methods train these [27]. Using these modules, the learner can get an attribute vector from the image: $\tilde{A}(y) = \mathbf{D_a}(\mathbf{E_i}(x))$.

In general, $\tilde{A}(y)$ will not match the imputed attributes $A'(y)$. Hypothetically replacing the i-th imputed attributes in $A'(y)$ with the image-derived counterpart in $\tilde{A}(y)$ would produce a new attribute description $A_i(y)$:

$$A_i(y) = \begin{cases} A'(y)[j] & j \neq i \\ \mathbf{D_a}(\mathbf{E_i}(x_y))[j] & j = i \end{cases} \quad (8)$$

Image-based strategy then picks the attribute j which maximally pushes the embedding of the hypothetical attribute vector $A_j(y)$ closest to the novel class image encoding:

$$Q_I(y|A'(y), x_y)[i] = ||\mathbf{E_a}(A_i(y)) - \mathbf{E_i}((x_y))||_2^{-1} \quad (9)$$

$$\pi_I(I(y), A'(y), x_y) = \arg\max_{j \notin I(y)} Q_I(y|A'(y), x_y)[j] \quad (10)$$

4. Results

4.1. Dataset and Implementation Details

For all experiments we use 2048-dimensional features from ResNet-101 [11]. We compare our method on three zero-shot benchmark datasets: CUB-200-2011 [35] (CUB), Animals with Attributes 2 [38] (AWA2) and the SUN attribute dataset [21] (SUN). CUB is annotated with 312 part-based attributes; AWA2 and SUN have 85 and 102 attributes respectively. These attributes are labelled for every class. We use the standard train-test split from [38] for all benchmarks.

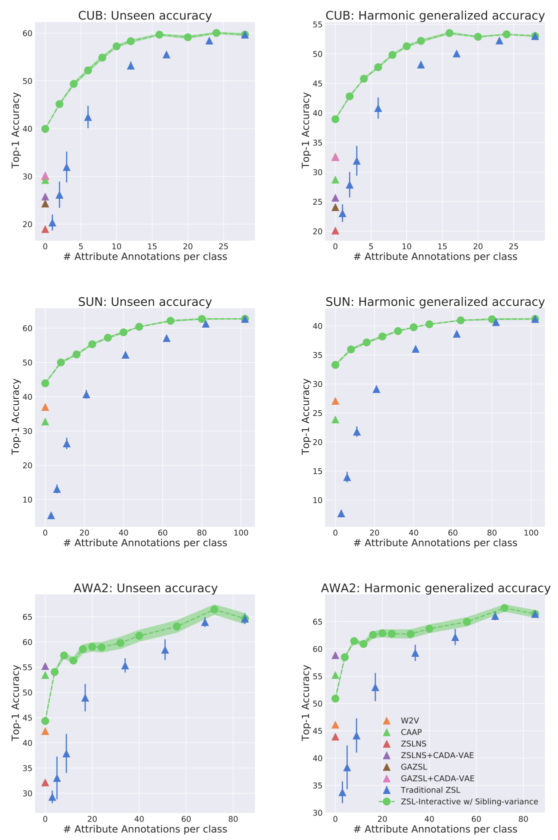

Figure 4. Performance of our method in terms of top-1 per-class classification accuracy vs. number of annotations provided by the annotators during deployment with CADA-VAE as base model. ZSL-Interactive performs better than the traditional ZSL on all benchmarks and better than unsupervised baselines on CUB and SUN, proving its effectiveness.

Expert annotations of most similar base class: Our problem setup requires that the annotator provide the most similar base class for each novel class. Since CUB is a specialized domain requiring bird expertise, we worked with a professional bird watcher to annotate this information for every novel class. AWA2 and SUN do not require expert knowledge, so we collected the "similar class" information using 3 annotators. We took a majority vote, and in cases when all 3 disagreed we asked them to come to a consensus. We will release these expert annotations publicly upon acceptance.

Taxonomy: For SUN, the taxonomy is already available along with the dataset. We manually created a taxonomy for AWA2 by looking at the family in biological nomenclature. For CUB, we use the general class name as the parent in the taxonomy (Hooded Oriole → Oriole).

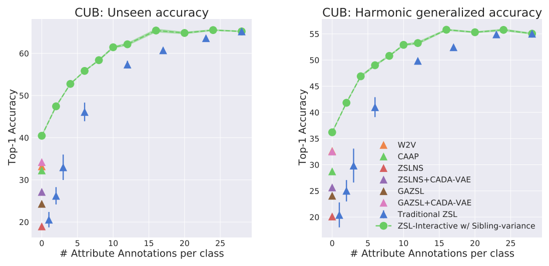

Figure 5. Performance of our method with TF-VAEGAN as base model. ZSL-Interactive performs better than the traditional ZSL and unsupervised baselines, proving the generalizability of ZSL-Interactive to other zero-shot models.

Base zero-shot learner: To show generalization of our method over different models we experiment with two base zero-shot learners: CADA-VAE[27] and TF-VAEGAN [19]. CADA-VAE trains two variational auto-encoders on the base classes to learn a common embedding space for attribute descriptions and images. It then trains novel class classifiers in this latent space. All the hyperparameters for the architecture and training are kept as prescribed by [27]. TF-VAEGAN uses a VAE-GAN [15] to generate realistic features from attributes and uses these generated features from unseen class to train a classifier. We show the results averaged over 6 different runs of the model.

Metrics: We measure the per-class classification accuracy over unseen classes and the harmonic mean of seen and unseen classification accuracy for generalized zero-shot learning. We plot these metrics as a function of the annotation budget for each approach.

4.2. Is interactive ZSL accurate and time-efficient?

We first ask how our proposed interactive setup compares with prior work on ZSL in terms of accuracy and the burden of annotation. We compare our proposed setup against the following baselines:

- **Traditional ZSL** with full attribute annotation. Reducing annotation effort here can only be achieved by using a smaller attribute vocabulary. Given an annotation budget, a correspondingly smaller subset of attributes is chosen uniformly at random, and a new ZSL model is retrained with the chosen attributes. In deployment, all classes must be described with this reduced vocabulary.
- **W2V**: This unsupervised ZSL approach uses word2vec embedding vectors of the classes instead of attribute vectors [1].
- **CAAP**: This approach uses word embeddings for the classes and the attributes to find the attribute vector for unseen classes [3].
- **ZSLNS [23] and GAZSL [43]**: use wikipedia articles instead of attributes. Since the original papers use older backbones for ZSL, we show the number for both the original model and using ZSLNS (or GAZSL) extracted descriptors with newer backbones (CADA-VAE and TF-VAEGAN).

While the last three are often deemed unsupervised, they do require either a large corpus of text with attributes and classes to learn word embeddings, or carefully curated text articles typically edited by experts, thus indirectly requiring significant expert time, which we aim to minimize.

Note that reported numbers for unsupervised ZSL approaches typically use a different class split that ensures that each novel class has a closely related base class. Instead, we use the proposed splits in the attribute-based ZSL benchmark [38]. We have used the original code provided by the authors and changed the splits wherever available.

We compare our best performing acquisition function (*Sibling-variance*), to the above baselines. We plot the accuracy as a function of the amount of annotation in Figure 4 (CADA-VAE) and Figure 5 (TF-VAEGAN). We find:

- **The expert annotated closest base class is extremely informative**. With just that one annotation, one can recover almost two-thirds of the performance of a full zero-shot learning system with access to hundreds of attributes. Note that using a randomly sampled class as the “most similar class” instead of the expert annotated one yields very poor accuracy ($<20\%$) indicating that the expert-provided information is critically important.
- **Our approach can dramatically reduce annotator burden**. On CUB and SUN, our approach with only a third of the annotations (just 10 interactions per class) is as good as traditional ZSL with *all* of the annotations.
- **Our approach generalizes to other ZSL models**. Our method performs better than the baselines even when TF-VAEGAN is used as the base model Figure 5. This shows that as new ZSL methods develop, our method should generalize and can be used out-of-the-box with them. See supplementary for TF-VAEGAN results on other datasets.
- **Partial attributes more informative than unsupervised ZSL**. While some methods perform better than our method with less information on AWA2, our method beats all the baselines on CUB and SUN. It is relatively hard to find discriminative information in corpora like wikipedia for fine-grained categories like those in SUN and CUB. This is the reason attribute-based systems are essential when performing ZSL in fine-grained domains. Just providing most similar class annotation (*i.e., the field guide approach*) turns out to be significantly more useful than using text or word embedding as attributes.

4.3. How do different acquistion functions perform?

Our method performs better than baselines primarily for two reasons, the field-guide interface and an intelligent acquisition function. We evaluate how different acquision functions perform with attribute annotation costs. Among the various query strategies we have proposed, in general, the best performing query strategy is *Sibling-variance*.

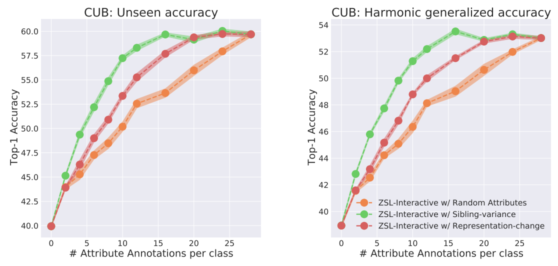

Figure 6. Performance of the two acquition functions with CADA-VAE on CUB. Both functions perform better than the random acquisition function. *Sibling-variance* performs better than *Representation-change*, but the latter does not require taxonomy information. See supplementary for results on other datasets.

Representation-change tends to have a weaker performance than *Sibling-variance* but it does not require the taxonomy to function and hence is still useful. Figure C.1 compares the 2 acquistion function against a random acquistion function as a baseline on CUB with CADA-VAE (see supplementary for other models and datasets). Both of our proposed query strategies far outperform random attribute selection as well as the traditional ZSL pipeline, *no matter what the labeling budget is*. This is true both for the unseen-only evaluation as well as the general evaluation.

4.4. Can we do better with images?

Figure D.1 shows the performance of our approach when one image is given by the annotator along with the interactive attributes annotations for CUB. Almost all of our conclusions from the previous section carry over, with the exception that *Representation-change* starts weaker than the baseline of choosing attributes to query randomly, but performs better than it in the later stages. Additionally, note that *Image-based* performs on par with the *Sibling-variance* without using any additional taxonomy information. This suggests that using the inconsistency between image-derived and imputed attributes to determine what attributes are useful is a viable approach to interact with annotators. See supplementary for results on other benchmarks.

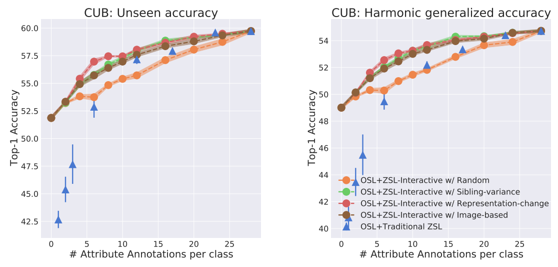

Figure 7. Performance of our method, zero+one-shot setting, where the annotator provides a single image for the novel class along with the interactively queried attributes. All methods perform better than the baselines; *Image-based* performs on par with *Sibling-variance* without requiring an additional taxonomy over the base classes.

4.5. Are acquisition functions better than experts?

One might question how our interactive method compares to collecting information from an expert. As discussed in section 1 a field-guide provides what the expert finds to be distinctive attribute differences without any interaction. We evaluate how this compares to the learner actively querying.

To this end, for 20 of the 50 novel classes the CUB dataset, we additionally ask an expert to identify the 10 most informative attributes that distinguish each novel class from its most similar base class, in order of importance. We use this infomation along with the similar class to construct the expert attribute baseline.

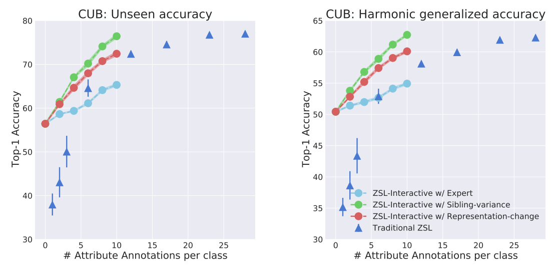

Figure 8. Performance of interactive methods against the expert selected attributes for CUB (20 novel classes). The learner selecting attributes performs better than the expert providing the attributes.

We find that the interactive methods are significantly better at asking for useful attributes than what the domain expert gives (Figure 8). This is a surprising result, as it shows that zero-shot learners learn about the domain differently than a human expert. In the set selected by experts, the experts give importance to attributes like “bird size” and “eye color”. While this information is useful to classify and differentiate between birds for a bird watcher, the model finds it difficult to understand the size of the bird as it is a relative property. Other attributes like “eye color” are very small in images (or obstructed) so the network cannot necessarily utilize that knowledge during representation learning. Hence, providing these attributes to the network is not very informative and it is better to let the learner select informative attributes.

4.6. Is the query selection good?

We look at the types of attributes queried by our method *Sibling-variance* for different parent categories of CUB. The attributes queried should vary based on the sibling classes. Figure B.1, shows the top-3 attributes queried first by the *Sibling-variance* method for Swallows and Cuckoos. It also shows the top-3 attributes picked by measuring variance over all classes. While bill length and overall shape have high variance over all classes, within Swallows (and within Cuckoos) the variance is not big and therefore should not be annotated first. For swallows, it can be seen from the image that throat, crown and forehead color varies within category, and hence should be acquired first. For Cuckoo, underpart,

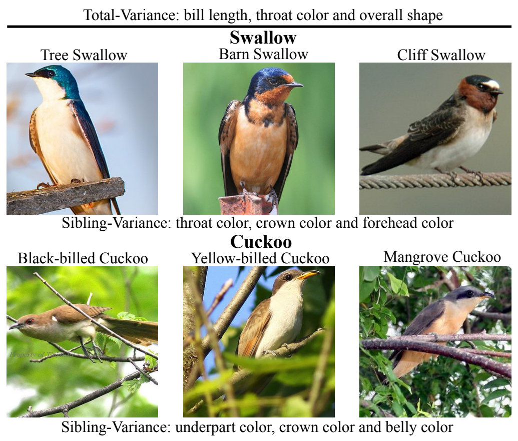
Total-Variance: bill length, throat color and overall shape
Swallow
Tree Swallow
Barn Swallow
Cliff Swallow
Sibling-Variance: throat color, crown color and forehead color
Cuckoo
Black-billed Cuckoo
Yellow-billed Cuckoo
Mangrove Cuckoo
Sibling-Variance: underpart color, crown color and belly color

Figure 9. Attributes selected by *Sibling-variance* for a parent class in the taxonomy and the attributes selected by measuring variance over all classes (top). As can be seen from the images within the categories, overall shape and bill length do not change much and measuring variance to pick attributes across all classes is not informative and our approach of looking at the siblings is necessary.

crown and belly color varies (See supplementary for other examples and datasets).

We also visualize how learning progresses with interactive questions and reponses. Figure 10 shows the t-SNE visualization of 2 CUB classes and the most similar base classes. The full attributes descriptor (bigger dot) for a class is surrounded by images of that class represented by smaller dots of the same color. Dots with red and black edges show the progression of novel class attributes as the learner interactively gains more information using the *Sibling-variance* and *random* acquisition function respectively. We see that *Sibling-variance* yields a faster progression from the similar base class attribute to the novel class. For example, the progression of the attribute descriptor encoding using *Sibling-variance* for “Tree Swallow” reaches the full attribute descriptor quicker, whereas with the *random* function the attribute is still close to the base class “Cliff Swallow”. More examples are in the supplementary.

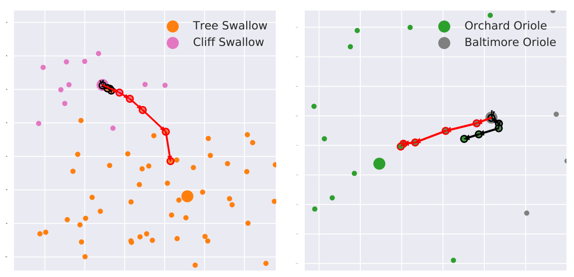

Figure 10. t-SNE visualization for 2 novel CUB classes (first class in legend) and its closest base classes (second) in the latent space of CADA-VAE. Smaller dots represent test images and larger dots represent class attribute embedding. Red edges show the progression of novel class attributes as learners interact using *Sibling-variance*. Dots with black edges show the progression with *random* function.

4.7. How much does the taxonomy help?

As seen from the results in previous sections, *Sibling-variance* lets us select informative attributes. In this section we evaluate if having a taxonomy is a requirement for this to work. Rather than measuring variance within the sibling classes we measure variance over all classes. This variant will always prioritize some types of attributes over others irrespective of the class, and ignore local variations within siblings. Therefore, we expect it will perform worse than with known taxonomy information.

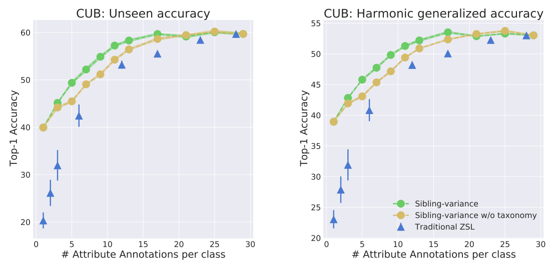

Figure 11. Comparing *Sibling-variance* against a variant where the taxonomy is unknown on CUB. The model loses accuracy if sibling classes are not used to measure *Sibling-variance*.

Figure F.1 compares the method without taxonomy information against the model where the taxonomy is known. As expected, this method loses performance because the local variation of a class cannot be measured, and hence those attributes cannot be selected. But even without the taxonomy the method performs better than ZSL and is useful for cases when the taxonomy is not known or difficult to acquire (See supplementary for performance on other datasets.). Finally, we also show that our methods are not very sensitive to similar base class selection as long as the expert chooses a class that is not wildly different looking (see supplementary).

5. Conclusions

In this work we show that an interactive *field-guide-inspired* annotation approach identifies informative attribute queries, and can achieve high performance by judiciously using an annotation budget. We present different ways to acquire informative sparse annotations from an annotator and show that is is better to let the machine choose the attributes to ask for (even when compared to an expert from the domain). Given these promising results, there are many avenues of future work: can one get even more cost-efficient by choosing a different number of attributes for different classes based on class confidence? We also need to bridge the gap between the attribute understanding of humans and machines, as our results show that human experts and neural-networks do not find the same attributes equally useful.

Acknowledgements. We thank our funding agencies TCS, NSF 1900783, and DARPA LwLL program (HR001118S0044) for their support.

References

[1] Zeynep Akata, Scott Reed, Daniel Walter, Honglak Lee, and Bernt Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, 2015. 1, 2, 6

[2] Ziad Al-Halah and Rainer Stiefelhagen. Automatic discovery, association estimation and learning of semantic attributes for a thousand categories. In *CVPR*, 2017. 1, 2

[3] Ziad Al-Halah, Makarand Tapaswi, and Rainer Stiefelhagen. Recovering the missing link: Predicting class-attribute associations for unsupervised zero-shot learning. In *CVPR*, 2016. 1, 2, 6

[4] Steve Branson, Pietro Perona, and Serge Belongie. Strong supervision from weak annotation: Interactive training of deformable part models. In *ICCV*, 2011. 3

[5] Steve Branson, Grant Van Horn, Catherine Wah, Pietro Perona, and Serge Belongie. The ignorant led by the blind: A hybrid human–machine vision system for fine-grained categorization. *IJCV*, 2014. 3

[6] Soravit Changpinyo, Wei-Lun Chao, Boqing Gong, and Fei Sha. Synthesized classifiers for zero-shot learning. In *CVPR*, 2016. 2

[7] Mohamed Elhoseiny, Babak Saleh, and Ahmed Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *ICCV*, 2013. 1, 2

[8] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *NeurIPS*, 2013. 1, 2

[9] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 2

[10] Joshua Goodman. Exponential priors for maximum entropy models. In *HLT-NAACL*, 2004. 2, 5

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[12] Elyor Kodirov, Tao Xiang, and Shaogang Gong. Semantic autoencoder for zero-shot learning. In *CVPR*, 2017. 2

[13] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *TPAMI*, 36(3):453–465, 2013. 2

[14] Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. Zero-data learning of new tasks. In *AAAI*, 2008. 2

[15] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016. 6

[16] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, et al. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *CVPR*, 2015. 1, 2

[17] David D Lewis and William A Gale. A sequential algorithm for training text classifiers. 1994. 2

[18] Ishan Misra, Ross Girshick, Rob Fergus, Martial Hebert, Abhinav Gupta, and Laurens van der Maaten. Learning by Asking Questions. In *CVPR*, 2018. 3

[19] Sanath Narayan, Akshita Gupta, Fahad Shahbaz Khan, Cees GM Snoek, and Ling Shao. Latent embedding feedback and discriminative features for zero-shot classification. *ECCV*, 2020. 6

[20] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S Corrado, and Jeffrey Dean. Zero-shot learning by convex combination of semantic embeddings. In *ICLR*, 2013. 2

[21] Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012. 5

[22] Genevieve Patterson, Grant Van Horn, Serge Belongie, Pietro Perona, and James Hays. Bootstrapping fine-grained classifiers: Active learning with a crowd in the loop. In *NeurIPS*, 2013. 3

[23] Ruizhi Qiao, Lingqiao Liu, Chunhua Shen, and Anton Van Den Hengel. Less is more: zero-shot learning from online textual documents with noise suppression. In *CVPR*, 2016. 1, 2, 6

[24] Marcus Rohrbach, Michael Stark, György Szarvas, Iryna Gurevych, and Bernt Schiele. What helps where–and why? semantic relatedness for knowledge transfer. In *CVPR*, 2010. 1, 2

[25] Bernardino Romera-Paredes and Philip Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015. 2

[26] Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In *ISIDA*, 2001. 2

[27] Edgar Schonfeld, Sayna Ebrahimi, Samarth Sinha, Trevor Darrell, and Zeynep Akata. Generalized zero-and few-shot learning via aligned variational autoencoders. In *CVPR*, pages 8247–8255, 2019. 2, 5, 6

[28] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *CoRR*, 2017. 2

[29] Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In *NeurIPS*, 2008. 2, 5

[30] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. *ICCV*, 2019. 2

[31] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. In *NeurIPS*, 2013. 2

[32] Yao-Hung Hubert Tsai and Ruslan Salakhutdinov. Improving one-shot learning through fusing side information. *CoRR*, 2017. 2, 5

[33] Vinay Kumar Verma and Piyush Rai. A simple exponential family framework for zero-shot learning. In *ECML*, 2017. 2

[34] Catherine Wah and Serge Belongie. Attribute-based detection of unfamiliar classes with humans in the loop. In *CVPR*, 2013. 3

[35] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5

[36] Catherine Wah, Grant Van Horn, Steve Branson, Subhransu Maji, Pietro Perona, and Serge Belongie. Similarity comparisons for interactive fine-grained categorization. In *CVPR*, 2014. 3

[37] Shuo Wang, Yuexiang Li, Kai Ma, Ruhui Ma, Haibing Guan, and Yefeng Zheng. Dual adversarial network for deep active learning. 2020. 2

[38] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *TPAMI*, 2018. 5, 6

[39] Yongqin Xian, Saurabh Sharma, Bernt Schiele, and Zeynep Akata. f-vaegan-d2: A feature generating framework for any-shot learning. In *CVPR*, 2019. 2

[40] Lin Yang, Yizhe Zhang, Jianxu Chen, Siyuan Zhang, and Danny Z Chen. Suggestive annotation: A deep active learning framework for biomedical image segmentation. In *ICMICCAI*, 2017. 2

[41] Aron Yu and Kristen Grauman. Thinking outside the pool: Active training image creation for relative attributes. In *CVPR*, 2019. 2

[42] Beichen Zhang, Liang Li, Shijie Yang, Shuhui Wang, Zheng-Jun Zha, and Qingming Huang. State-relabeling adversarial active learning. In *CVPR*, 2020. 2

[43] Yizhe Zhu, Mohamed Elhoseiny, Bingchen Liu, Xi Peng, and Ahmed Elgammal. A generative adversarial approach for zero-shot learning from noisy texts. In *CVPR*, 2018. 1, 2, 6

Supplementary Material

A. Overview

In this supplementary material we look at some more results that could not be presented in the main paper. Our code can be found at github.com/utkarshmall13/Field-Guide-ZSL We show attributes queried by *Sibling-variance* on SUN and AWA2 in Sec. B. Sec. C shows the performance of the two acquisition functions on AWA2 and SUN. In Sec. D we present results for when the interactive learner uses a single image for SUN and AWA2. In Sec. E we compare the performance of our method with TF-VAEGAN on the AWA2 and SUN. Sec. F shows the effect of not using a taxonomy in *Sibling-variance* for AWA2 and SUN. In Sec. G we show the learner's behavior when classes other than the annotators first choice are chosen for SUN and AWA2. Sec. H presents more t-SNE visualization examples of the learning progression for novel class descriptors on all three datasets. We also **strongly** encourage the reader to refer to the **supplementary video** for better visualizations of the t-SNE progression.

B. More Qualitative Evaluation

Figure B.1, shows the attributes queried first by the *Sibling-variance* method for 2 supercategories of SUN and AWA2. It also shows the attributes picked by measuring variance over all the classes. For SUN, attributes like "enclosed/open area" or "man-made/natural" may help in disambiguating between very different classes, but within a supercategory they do not help. For example, for the superclass "Indoor sports and leisure", all the classes are closed and man-made. But attributes like "competing", "spectating" are more informative. Similarly for indoor workplaces, attributes like "using tools" and "studying/learning" are very informative. Similar patterns can be seen on AWA2. *Sibling-variance* asks for attributes informative within the superclass.

C. Comparison of Acquisition Functions on AWA2 and CUB

Figure C.1 shows the performance of the two attribute querying acquisition functions with the CADA-VAE model on AWA2 and SUN. Our acquisition functions perform significantly better than a random acquisition function, showing the value of our field-guide annotation.

While the results for fine-grained dataset such as SUN are similar to that on CUB, for AWA2 *Representation-change* performs better than *Sibling-variance* in the later stages. This might be because the AWA2 model is trained for fewer coarse-grained classes with thousands of images and has a better representation and understanding of changing representation.

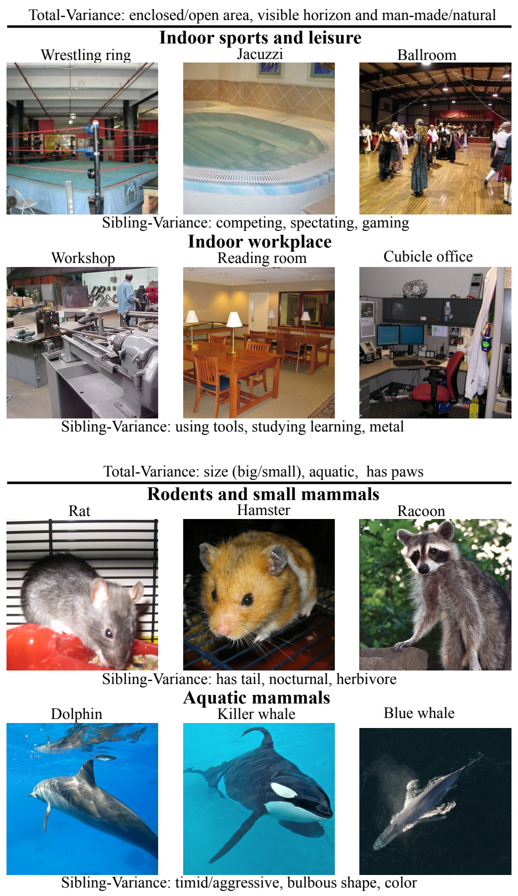
Total-Variance: enclosed/open area, visible horizon and man-made/natural
Indoor sports and leisure
Wrestling ring
Jacuzzi
Ballroom
Sibling-Variance: competing, spectating, gaming
Indoor workplace
Workshop
Reading room
Cubicle office
Sibling-Variance: using tools, studying learning, metal
Total-Variance: size (big/small), aquatic, has paws
Rodents and small mammals
Rat
Hamster
Racoon
Sibling-Variance: has tail, nocturnal, herbivore
Aquatic mammals
Dolphin
Killer whale
Blue whale
Sibling-Variance: timid/aggressive, bulbous shape, color

Figure B.1. Attributes selected by *Sibling-variance* for a parent class in the taxonomy and the attributes selected by measuring variance over all classes (top) for SUN and AWA2.

D. *Image-based* Results for AWA2 and SUN

Figure D.1 shows the performance of our approach when one image is given by the annotator along with the interactive attribute annotations for AWA2 and SUN. As for CUB, all the methods perform better than the baselines. For SUN, the *Image-based* function performs on par with *Sibling-variance* without requiring an additional taxonomy. For AWA2, the *Image-based* function performs better than all the methods we propose and the baselines. AWA2 has more training images and the classes are not very fine-grained. This might be the reason why *Image-based* acquisition functions work better for AWA2.

E. Performance of TF-VAEGAN on AWA2 and SUN

Figure E.1 show the performance of our approach when the base model is TF-VAEGAN on SUN and AWA2. Our field-guide way of annotation works better then traditional ZSL baselines for both the dataset, proving the effectiveness and generalization of our method. For fine-grained classes such as SUN and CUB, our method is .

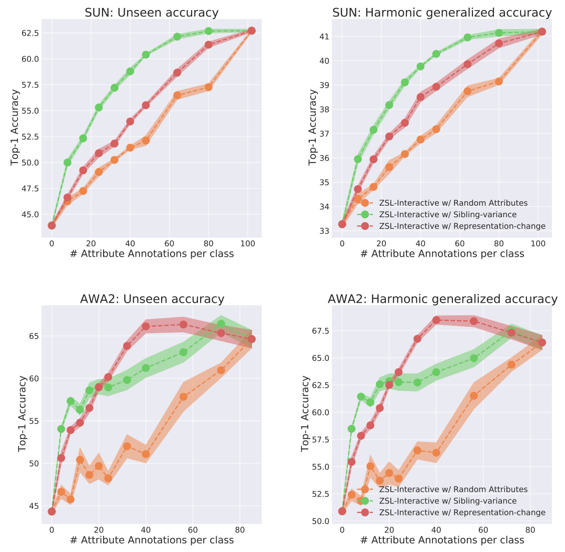

Figure C.1. Performance of the two acquisition functions with CADA-VAE on AWA2 and SUN. Both functions perform better than the random acquisition function. On SUN, *Sibling-variance* performs better than *Representation-change*, but the latter does not require taxonomy information. Results on AWA2 are different, in the earlier stages *Sibling-variance* is better than *Representation-change*, but in the later stages *Representation-change* is better.

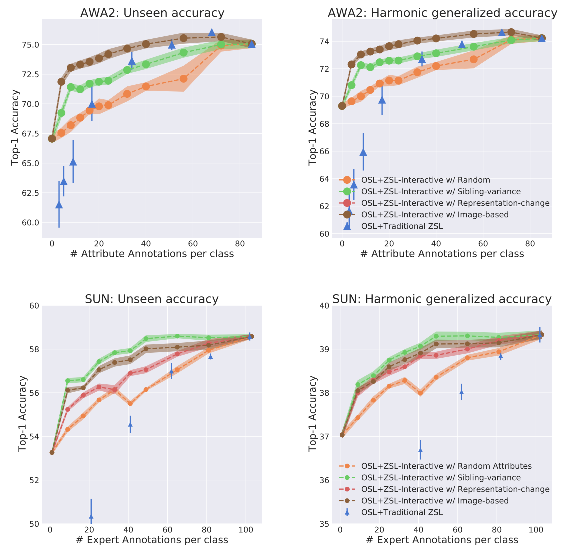

Figure D.1. Performance of our method under the zero+one-shot setting when the annotator provides a single image for novel class along with the interactive attribute values for SUN and AWA2.

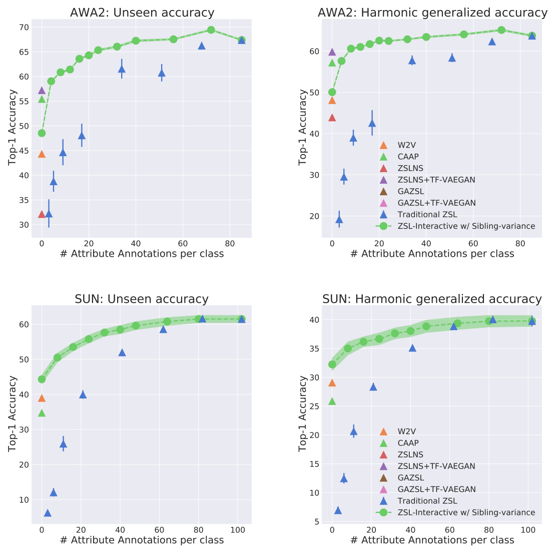

Figure E.1. Comparison of our method against unsupervised and traditional ZSL baselines with TF-VAEGAN as the base model. Our method performs better than tradtional ZSL at the same attribute annotation cost for both AWA2 and SUN. Similar to results for CADA-VAE in the main paper, our method works better than the unsupervised baselines for SUN.

F. Performance on AWA2 and SUN Without Taxonomy

Figure F.1 compares the method without taxonomy information against the model where the taxonomy is known. The results follow the conclusion from the main paper. When there is no taxonomy information available, this method loses performance because the local variation of a class cannot be measured, and hence those attributes cannot be selected. But even without the taxonomy the method performs better than ZSL and is useful for cases when the taxonomy is not known or difficult to acquire.

G. Effect on Performance When Changing Similar Base Class

The similar class given by the annotator is certainly more important than each attribute annotation. We look at the effect of choosing another class: either a random class from the full set, or a sibling of the expert selection that is closest to the expert selected sibling in word2vec embedding space.

Figure G.1 shows *Sibling-variance* with these annotations along with the ZSL baseline on CUB. The similar class chosen by annotators performs best. When we choose a class that is close to this (sibling), the method performs slightly worse. This shows that although our method performance is affected if a non-optimal nearest class is chosen, it is not very sensitive to it. Both these variants do significantly better than randomly selecting a similar class. This suggests that

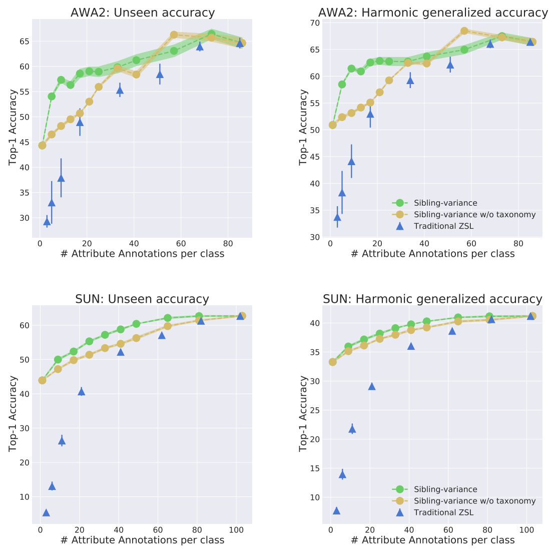

Figure F.1. Comparing *Sibling-variance* against a variant where the taxonomy is unknown on AWA2 and SUN. The model loses accuracy if sibling classes are not used to measure *Sibling-variance*.

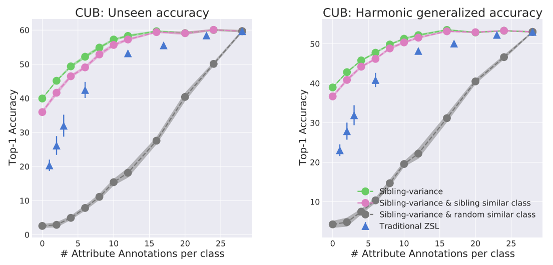

Figure G.1. Comparing different variants for selecting the similar class $S(y)$ on CUB dataset. The model is not sensitive to the similar class as long as the selected class is not wildly different.

the interactive model will perform well as long as annotators do not provide a wildly different looking similar class.

Figure G.2 shows the results for *Sibling-variance* on AWA2 and SUN. The similar class chosen by annotators performs similar to when a sibling base class is chosen for AWA2 and SUN. The performance is again not very sensitive to choosing the similar class as long as they are not very different.

Along with the sensitivity to the choice of the similar class, we also evaluated our method with sensitivety to attribute values. Note that incorrect or noisy attribute values will affect not just our proposed active ZSL but also the traditional non-interactive ZSL. With 10% noise in the novel attributes, when all attributes are provided, both our method and traditional ZSL see a $\sim 3\%$ drop in performance. With partial attribute annotations (5 per class), our proposed annotation strategy ($\sim 1\%$ drop) fares much better than traditional ZSL annotation ($\sim 5\%$ drop).

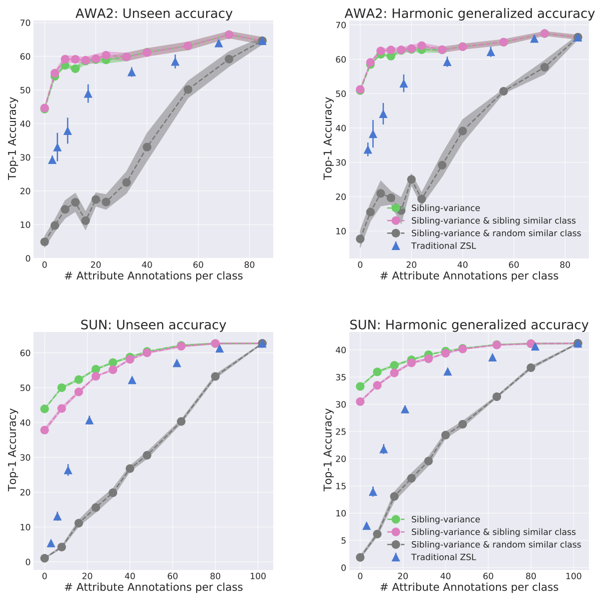

Figure G.2. Comparing different variants for selecting the similar class $S(y)$ on AWA2 and SUN dataset. The model is not sensitive to the similar class as long as the selected class is not wildly different.

H. t-SNE Visualizations for More Classes and Dataset

Figure H.1 show the progression of *Sibling-variance* and random attributes for all 10 AWA2 novel classes. Note that in the standard split of AWA2, the classes are split in a way that sometimes no good similar classes could be found. For example, both seal and walrus are in the test split and hence the annotators chose beaver and walrus as similar classes. Similarly no good base class is there for giraffe and bat so the annotators had to chose zebra and squirrel. Nonetheless the faster progression towards novel classes' images and attributes can be seen for the classes when using *Sibling-variance* over random attributes.

Figure H.2 and H.3 show the progression of *Sibling-variance* and random attributes for all 20 novel classes of CUB (out of 50) and SUN (out of 70). Faster progression can be seen for *Sibling-variance* over these classes as well.

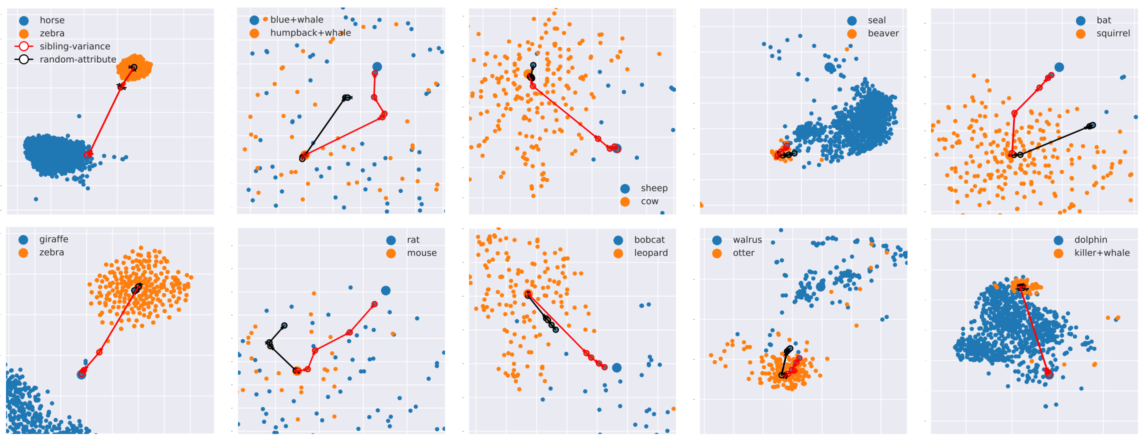

Figure H.1. t-SNE visualizations. For all 10 AWA2 novel classes and corresponding similar base classes. Smaller dots represent test images and larger dots represent class attribute embeddings. Red edges show the progression of novel class attributes as learners interact using *Sibling-variance*. Dots with black edges show the progression with the *random* function. Both methods start at the base class attribute descriptor, and aim to reach to the novel class descriptor with as few interactions as possible. In most cases *Sibling-variance* reaches closer to the novel class descriptor quicker in contrast to *random*.

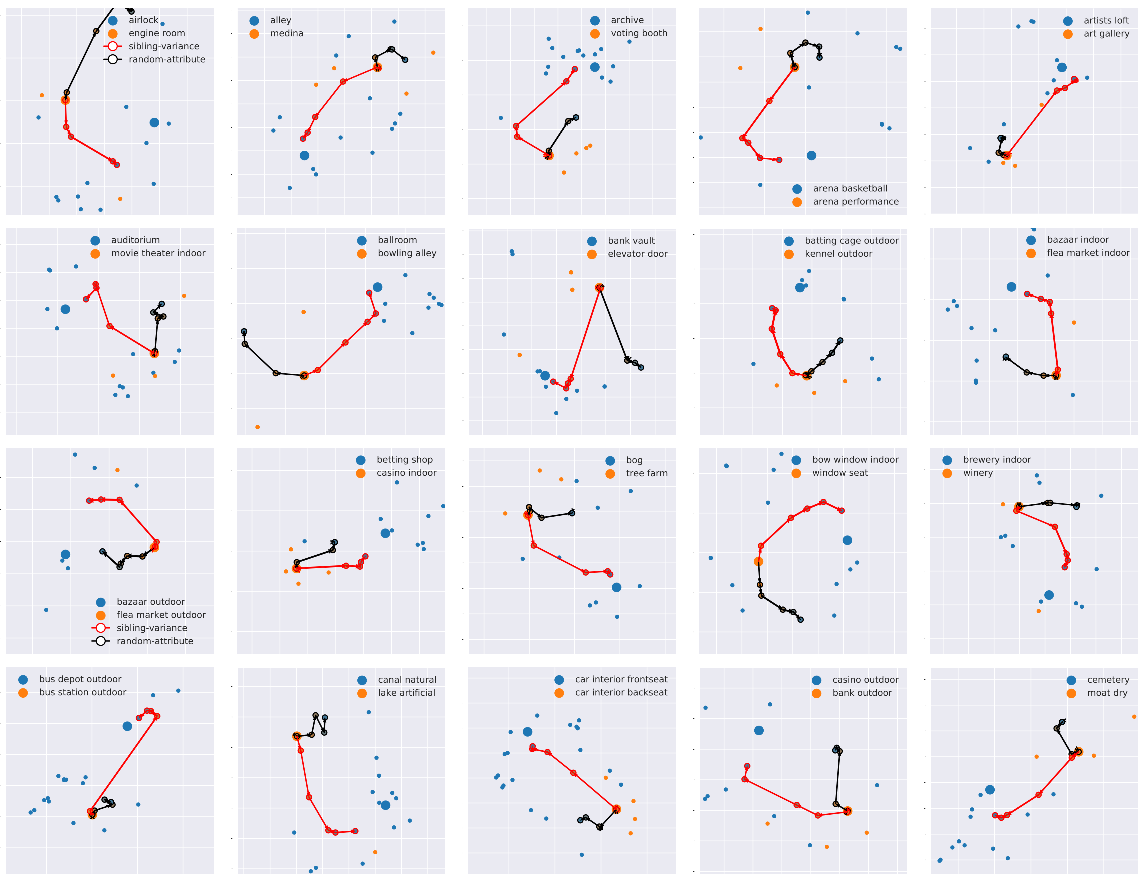

Figure H.2. t-SNE visualizations. For 20 SUN novel classes and corresponding similar base classes. Smaller dots represent test images and larger dots represent class attribute embeddings. Red edges show the progression of novel class attributes as learners interact using *Sibling-variance*. Dots with black edges show the progression with the *random* function. Both methods start at the base class attribute descriptor, and aim to reach the novel class descriptor with as few interactions as possible. In most cases *Sibling-variance* reaches closer to the novel class descriptor quicker in contrast to *random*.

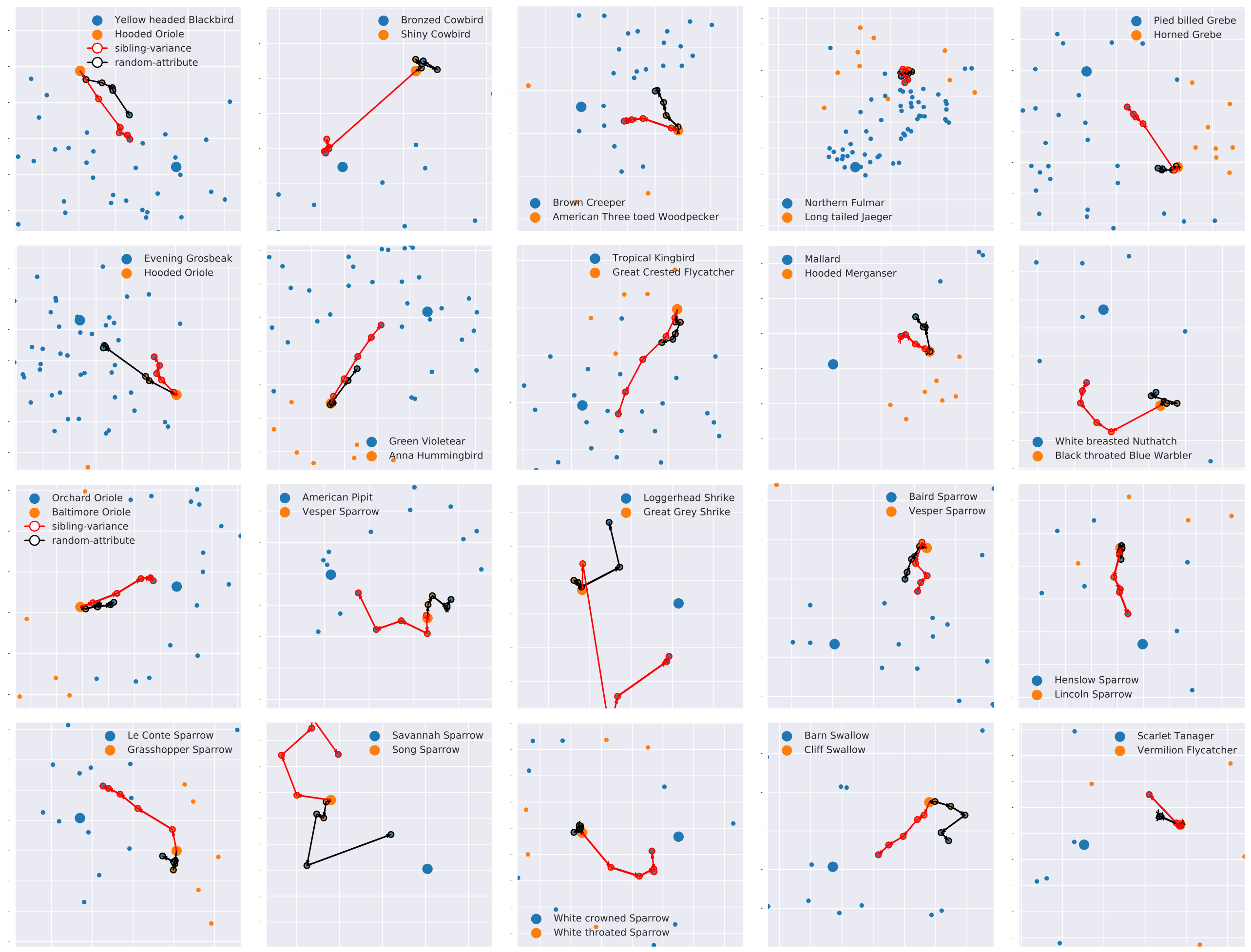

Figure H.3. t-SNE visualizations. For 20 CUB novel classes and corresponding similar base classes. Smaller dots represent test images and larger dots represent class attribute embeddings. Red edges show the progression of novel class attributes as learners interact using *Sibling-variance*. Dots with black edges show the progression with the *random* function. Both methods start at the base class attribute descriptor, and aim to reach the novel class descriptor with as fewer interactions as possible. In most cases *Sibling-variance* reaches closer to the novel class descriptor quicker in contrast to *random*.