

Fine-Grained Visual Comparisons

PhD Dissertation

by

Aron Yingbo Yu

Department of Electrical and Computer Engineering
University of Texas at Austin

Committee:

Dr. Kristen Grauman

Dr. Brian Evans

Dr. Alan Bovik

Dr. Qixing Huang

Dr. Derek Hoiem

Abstract

Beyond recognizing objects, a computer vision system ought to be able to *compare* them. A promising way to represent visual comparisons is through *attributes*, which are mid-level properties that appear across category boundaries. The ability to compare attributes opens up new opportunities in areas such as online shopping, object recognition, and human biometrics, where a relative decision is often more informative than its binary counterpart. For example, given two human faces, a decision that one face is *smiling* more than the other may be more informative—and even more appropriate—than a hard yes or no decision.

In this thesis, I explore the task of fine-grained visual comparisons, or *relative attributes*. Given two images, we want to predict which exhibits a particular visual attribute more than the other. Specifically, I explore the scenario where the images exhibit subtle—thus *fine-grained*—visual differences. I propose improvements on two fronts, through the algorithms and the source data, to target these fine-grained comparison tasks that standard models fail to handle.

On the algorithmic front, existing relative attribute methods rely exclusively on global ranking functions. However, rarely will the visual cues relevant to a comparison be constant for all data, nor will humans’ perception of the attribute necessarily permit a global ordering. Furthermore, not every image pair is even orderable for a given attribute. Attempting to map relative attribute ranks to “equality” predictions is non-trivial, particularly since the span of indistinguishable pairs in attribute space may vary in different parts of the feature space. To address these issues, we introduce *local learning* approaches for fine-grained visual comparisons, where a predictive model is trained on-the-fly using only the data most relevant to the novel input. In particular, given a novel pair of images, we develop local learning methods to (1) infer their relative attribute ordering with a ranking function trained using only analogous labeled image pairs, (2) infer the optimal “neighborhood”, i.e., the subset of the training instances most relevant for training a given local model, and (3) infer whether the pair is even distinguishable, based on a local model for *just noticeable differences* in attributes.

On the source data front, we address the *sparsity of supervision* issue that affects all ranking algorithms for fine-grained tasks. Due to the pairwise nature of the

supervision labels, the space of all possible comparisons is quadratic with respect to the total number of images. Even if we could hypothetically obtain complete supervision, we still cannot ensure sufficient diversity of fine-grained differences, at least not using only the provided real images. The problem is that we lack a direct way to curate photos demonstrating all sorts of subtle attribute changes.

We propose to overcome this challenge using synthetic images that are conditionally generated based on the strength of a set of attributes. Building on a state-of-the-art image generation engine, we generate pairs of training images both passively and actively. In the passive case, we sample pairs of pre-generated training images exhibiting slight modifications of individual attributes. The proposed “semantic jitter” approach can be seen as a new form of data augmentation where training samples with subtly different attributes are automatically created. In the active case, we jointly learn the attribute ranking task while also learning to generate realistic image pairs that will benefit that task. We introduce an end-to-end framework that dynamically “imagines” image pairs that would confuse the current model, presents them to human annotators for labeling, then improves the predictive model with the new examples. Whether generated actively or passively, we augment real training image pairs with these generated pairs, and then train attribute ranking models to predict the relative strength of an attribute in novel pairs of real images. Our results demonstrate the effectiveness of bootstrapping imperfect image generators to counteract supervision sparsity in learning-to-rank models.

Overall, our proposed methods outperform state-of-the-art baselines for relative attribute prediction on challenging datasets, including UT-Zap50K, a large new shoe dataset curated specifically for fine-grained comparison tasks. We find that for fine-grained comparisons, performance is optimized when *the algorithm works in conjunction with the data source*. In this thesis, the optimal pipeline functions by first densifying the attribute space through generating the “right” data, and then applying fine-grained algorithms that leverage and learn from these “right” data.

Table of Contents

Abstract	ii
Chapter 1. Introduction	1
Chapter 2. Related Work	8
2.1 Attribute Comparison	8
2.2 Fine-Grained Visual Tasks	9
2.3 Local Learning	10
2.4 Attribute and Image Synthesis	11
2.5 Learning using Synthetic Images	12
2.6 Active Learning	13
Chapter 3. Datasets	14
3.1 UT-Zap50K: Fine-Grained Shoe Dataset	14
3.2 Improved UT-Zap50K with a Fine-Grained Lexicon	16
3.3 Face Datasets	19
3.4 Scene Dataset	21
Chapter 4. Local Ranking Functions for Attributes	22
4.1 Ranking Functions for Relative Attributes	23
4.1.1 Large-Margin Ranking Functions	23
4.1.2 Deep Ranking Functions	24
4.2 Approach	25
4.2.1 Local Learning for Visual Comparisons	26
4.2.2 Selecting Fine-Grained Neighboring Pairs	27
4.3 Evaluation	29
4.3.1 Experimental Setup	29
4.3.2 Zappos Results	32
4.3.3 Scenes and PubFig Results	33
4.4 Predicting Useful Neighborhoods	34
4.4.1 Generating Neighborhoods	36
4.4.2 Model Learning using Compressed Sensing	37
4.4.3 Inferring Neighborhoods	37
4.5 Discussion	38

Chapter 5. Just Noticeable Attribute Differences	39
5.1 Approach	41
5.1.1 Local Bayesian Model of Distinguishability	41
5.1.2 The Likelihood Model	42
5.1.3 The Prior Model	43
5.2 Evaluation	44
5.2.1 Experimental Setup	44
5.2.2 Just Noticeable Difference Detection	46
5.2.3 Image Search Application	48
5.3 Discussion	52
Chapter 6. Dense Supervision Through Semantic Jitter	54
6.1 Approach	55
6.1.1 Attribute-Conditioned Image Generator	56
6.1.2 Generating Dense Synthetic Image Pairs	57
6.1.3 Learning to Rank with Hybrid Comparisons	59
6.2 Evaluation	61
6.2.1 Experimental Setup	61
6.2.2 Fashion Images of Shoes	63
6.2.3 Human Faces	66
6.3 Discussion	67
Chapter 7. Active Training Image Creation	68
7.1 Approach	69
7.1.1 Ranking Module	70
7.1.2 Generator Module	70
7.1.3 Control Module	71
7.1.4 Training and Active Image Creation	72
7.2 Experiments	74
7.2.1 Experimental Setup	74
7.2.2 Pairwise Comparisons using ATTIC	76
7.2.3 Active vs. Passive Training Image Generation	78
7.2.4 Comparison to Previously Published Results	79
7.2.5 Qualitative Analysis	79
7.2.6 Insights into Synthetic Images	81
7.3 Discussion	82

Chapter 8. Conclusion	87
8.1 Extension Ideas	88
8.2 Future Work	89
Bibliography	90

Chapter 1

Introduction

Attributes are visual properties describable in words, capturing anything from material properties (*metallic, furry*), shapes (*flat, boxy*), expressions (*smiling, surprised*), to functions (*sittable, drinkable*). Since their introduction to the recognition community [28, 62, 66], attributes have inspired a number of useful applications in image search [13, 58, 59, 62, 105], biometrics [16, 51, 94], and language-based supervision for recognition [8, 66, 84, 104, 123].

Existing attribute models come in one of two forms: categorical or relative. Whereas categorical attributes are suited only for clear-cut predicates, such as *wooden* or *four-legged*, relative attributes can represent “real-valued” properties that inherently exhibit a spectrum of strengths, such as *serious* or *sporty*. These spectra allow a computer vision system to go beyond recognition into comparison. For example, with a model for the relative attribute *brightness*, a system could judge which of two images is *brighter* than the other, as opposed to simply labeling them as bright or not bright.

Attribute comparisons open up a number of interesting possibilities and applications. In biometrics, the system could interpret descriptions like, “the suspect is *taller* than him” [94]. In image search, the user could supply semantic feedback to pinpoint his desired content: “the shoes I want to buy are like these but *more masculine*” [59]. For object recognition, human supervisors could teach the system by relating new objects to previously learned ones, e.g., “a mule has a tail *longer than* a donkey’s” [8, 84, 104]. For subjective visual tasks, users could teach the system their personal perception, e.g., about which human faces are *more attractive* than others [2].

One typically learns a relative attribute in a learning-to-rank setting; training data is ordered (e.g., we are told image A has the attribute more than image B) and a ranking function is optimized to preserve those orderings. Given a new image, the function returns a score conveying how strongly the attribute is present [2, 14, 18, 27, 59, 69, 75, 84, 95, 96]. While a promising direction, the standard ranking approach tends

to fail when faced with **fine-grained visual comparisons**. In particular, the standard approach falls short on two fronts:

1. Model wise, the underlying algorithms cannot reliably predict comparisons when a novel pair of images exhibits subtle visual differences. It also does not permit equality prediction, where the novel pair of images is so similar that the difference is indistinguishable. The former scenario includes both the case where the images are globally similar, making all distinctions fine-grained, as well as the case where the images are very similar only in terms of the attribute of interest.
2. Data wise, the existing relative datasets suffer from the *sparsity of supervision* issue, for both label availability and image availability. Due to the pairwise nature of the training data, which results in a quadratic number of possibilities, the labeling process quickly becomes intractable. Furthermore, even when exhaustively labeled, the images within the dataset need not guarantee sufficient representation of fine-grained differences.

We contend that fine-grained comparisons are critical to get right, since this is where modeling relative attributes ought to have great power. Otherwise, we could just learn coarse categories of appearance (“bright scenes”, “dark scenes”) and manually define their ordering. In particular, fine-grained visual comparisons are valuable for sophisticated image search and browsing applications, such as distinguishing subtle properties between similar products in an online catalog, as well as analysis tasks involving nuanced perception, such as detecting slight shades of human facial expressions or distinguishing the identifying traits between otherwise similar-looking people. We use the term “fine-grained” to reflect that the visual differences we want to detect are slight; similarly, fine-grained recognition tasks in computer vision [11, 29, 74, 121] are also concerned with visual subtleties, though for the case of classifying subclasses.

Keeping the shortcomings above in mind, in this thesis, I explore the following approaches with the ultimate goal of improving performance on fine-grained visual comparison tasks.

Local Ranking Functions for Attributes: Why do existing global ranking functions experience difficulties when making fine-grained comparisons? The problem is

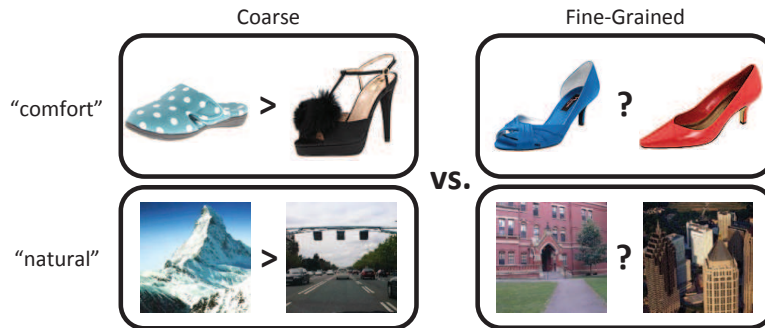


Figure 1.1: A global ranking function may be suitable for *coarse* ranking tasks, but *fine-grained* ranking tasks require attention to subtle details—and which details are important may vary in different parts of the feature space. The first major component of my thesis (Chap. 4) proposes a local learning approach to train comparative attributes based on fine-grained analogous pairs.

that while a single learned function tends to accommodate the gross visual differences that govern the attribute’s spectrum, it cannot simultaneously account for the many fine-grained differences among closely related examples, each of which may be due to a distinct set of visual cues. For example, what makes a slipper appear *more comfortable* than a high heel is different than what makes one high heel appear more comfortable than another; what makes a mountain scene appear *more natural* than a highway is different than what makes a suburb more natural than a downtown skyscraper (Fig. 1.1).

To increase the specificity of the ranker, I introduce *local learning* algorithms for fine-grained visual comparisons. Local learning is an instance of “lazy learning”, where one defers processing of the training data until test time. Rather than estimate a single global model from all training data, local learning methods instead focus on a subset of the data most relevant to the particular test instance. Simply put, we want to learn from the *right* data instead of with just *any* data. This approach helps learn fine-grained models tailored to the new input, and makes it possible to adjust the capacity of the learning algorithm to the local properties of the data [9]. The main idea of my approach is to identify analogous neighbors in a pairwise setting and to learn from attribute-specific features based on metric learning. This work appeared in CVPR 2014 [125], NIPS 2014 [126], and a book chapter [124].

Just Noticeable Differences: Having considered a new learning model to capture fine-grained differences, we next turn to the challenge of detecting whether two images exhibit a difference at all in the first place. What happens when the fine-grained differences between two images become so subtle that they become indistinguishable?



Figure 1.2: At what point is the strength of an attribute indistinguishable between two images? While existing relative attribute methods are restricted to inferring a total order, in reality there are images that look different but where the attribute is nonetheless perceived as “equally strong”. For example, in the fourth and fifth images of Obama, is the difference in *seriousness* noticeable enough to warrant a relative comparison? The second major component of my thesis (Chap. 5) addresses the just noticeable differences problem for attributes.

Existing attribute models assume that all images are orderable. In particular, they assume that *at test time*, the system can and should always distinguish which image in a pair exhibits the attribute more. To illustrate how this is problematic, imagine you are given a pile of images of Barack Obama, and you must sort them according to where he looks most to least *serious*. Surely there will be some obvious ones where he is more serious or less serious. There will even be image pairs where the distinction is quite subtle, yet still perceptible, thus fine-grained. However, you are likely to conclude that forcing a *total* order is meaningless: while the images exhibit different degrees of the attribute seriousness, at some point the differences become indistinguishable. It is not that the pixel patterns in indistinguishable image pairs are literally the same—they just cannot be characterized consistently as anything other than “equally serious” (Fig. 1.2).

To handle such equality cases, I introduce a Bayesian *just noticeable difference* model that learns from the local statistics of orderability. Just noticeable difference (JND) is a concept from psychophysics, loosely defined as the amount a stimulus has to be changed in order for it to be detectable by human observers at least half the time [50]. We adapt this idea for fine-grained comparisons by learning to predict the distinguishability of image pairs. Note that we borrow the term JND here only to capture our intent. The main idea of my approach is to account for non-uniformity in the attribute space using local statistics around the novel pair and to predict distinguishability in a probabilistic local learning manner. Such an approach is different from traditional forms of JND as it lacks a mechanism to modify the target stimulus (attribute strength in this

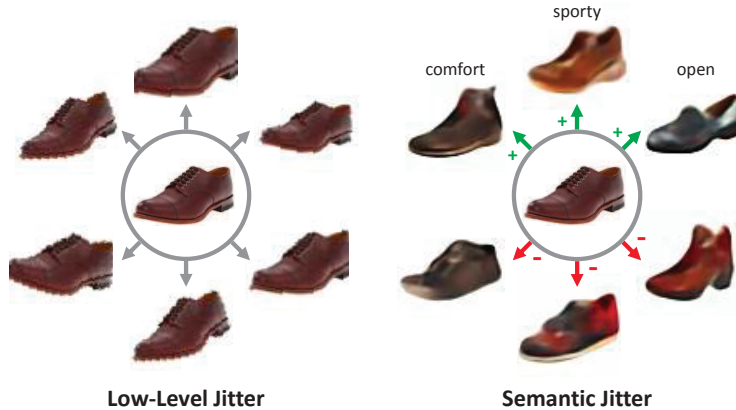


Figure 1.3: Whereas standard data augmentation with low-level “jitter” (left) expands training data with image-space alterations (mirroring, scaling, etc.), our *semantic jitter* (right) expands training data with high-level alterations, tweaking semantic properties in a controlled manner, both passively and actively. The third and fourth major components of my thesis (Chap. 6 and 7) propose to densify the attribute space using synthetically generated images.

case) incrementally in arbitrary small steps. This work appeared in ICCV 2015 [127] and a book chapter [124].

Dense Supervision through Semantic Jitter: Aside from algorithmic improvements, I also explore the orthogonal issue of supervision sparsity for fine-grained comparisons. Given a dataset, how do we ensure that the existing images sufficiently illustrate a diverse set of fine-grained differences? The underlying assumption of existing models (including our proposed local models above) is that the “right” data is already available in the training set, and that even if the existing labels are sparse, it is always possible to label more image pairs to increase the density of the overall supervision. However, this is not necessarily the case when training a fine-grained ranking model. The issue is that we lack a direct way to curate photos demonstrating all sorts of subtle attribute changes. For example, how might we gather unlabeled image pairs depicting all subtle differences in “sportiness” in clothing images or “surprisedness” in faces? As a result, even today’s best datasets contain only partial representations of an attribute.

To overcome this sparsity of supervision issue, I propose the addition of *synthetic image pairs* into the training data when learning any fine-grained ranking models. The idea is to synthesize plausible photos exhibiting variations along a given attribute from a generative model, thereby recovering samples in regions of the attribute space that are under-represented among the real training images. This can be seen as semantic “jitter-

ing” of the data to augment real image training sets with nearby variations. The systemic perturbation of images through label-preserving transforms like mirroring/scaling is now common practice in training deep networks for classification [25, 106, 116], and more recently for ranking [107, 108, 122] as well. Whereas such low-level image manipulations are performed independent of the semantic content of the training instance, the variations introduced by my approach are high-level changes that affect the very meaning of the image. In other words, our jitter has a semantic basis rather than a purely geometric/photometric basis (Fig. 1.3). This work appeared in ICCV 2017 [128].

Active Training Image Creation: Finally, I take this sparsity of supervision issue one step further by learning to generate these synthetic image pairs in an active manner. While traditional *active learning* methods can try to prioritize informative instances for labeling, the curation problem remains: existing active selection methods scan the pool of manually curated unlabeled images when choosing which ones to label [32, 113, 114, 135]. Our idea is for a system to directly synthesize image pairs that would confuse the current ranking model, then present them to human annotators for labeling. While my semantic jittering approach above generates individual images passively by adjusting one target attribute at a time, my active training approach here generates pairs of images by adjusting multiple attributes simultaneously in an adversarial manner.

To this end, I propose an end-to-end deep network consisting of an attribute-conditioned image generator, a ranking network, and a *control* network. The control network learns to generate latent visual parameters to present to the image generator so as to form image pairs that would be difficult for the ranker. Thus, rather than passively generate synthetic image pairs offline based on heuristics, the control module learns a *function* to create novel instances, potentially improving the exploration of the relevant image space. We train the ranker and controller in an adversarial manner, and we solicit manual labels for the resulting images in batches. As a result, the set of synthesized training instances continues to evolve, as does the ranker. This work appears in [129].

To summarize, in this thesis, I address fundamental issues on both the algorithm and the data fronts for fine-grained visual comparison problems. In particular, I propose (1) to learn local models tailored to each novel comparison at hand [125, 126], (2) to allow equality predictions based on local distinguishability statistics [127], and (3) to densify under-represented regions of the attribute space using synthetic image pairs, both passively [128] and actively [129].

Lastly, to complement these new approaches, I also curate and crowdsource labels for a brand new large-scale shoe dataset, the UT Zappos50K (**UT-Zap50K**), which is tailor made for fine-grained comparison tasks. We contribute to this dataset over two works, collecting over 45,000 pairwise labels across 11 relative attributes. This is to-date the largest relative attribute dataset that includes instance-level supervision.

Roadmap: The rest of this dissertation proceeds as follows. In Chapter 2, we discuss related work in the areas of relative attributes, local learning, fine-grained visual learning, image synthesis, and active learning. In Chapter 3, we present the various datasets used throughout this thesis. In Chapters 4 and 5, we discuss in detail our proposed approaches for fine-grained visual comparisons and equality prediction. In Chapter 6 and 7, we present new methods to improve ranking models in general by addressing the sparsity of supervision issue. Finally, we conclude in Chapter 8 with an overview of potential future work beyond this thesis.

Chapter 2

Related Work

In this chapter, I review the related work to the research that will be presented in Chapters 4, 5, 6, and 7. These literature provide a context for our proposed methods and serve as a starting point for readers regarding the topics covered in this thesis. Specifically, I address the areas of relative attributes, fine-grained local learning, image generation, learning from synthetic images, and active learning.

2.1 Attribute Comparison

Since the introduction of *relative attributes* [84], the task of attribute comparisons has gained attention for its variety of applications, such as online shopping [59, 60], biometrics [93], novel forms of low-supervision learning [8, 104], font selection [81], 3D model editing [15, 130], and visual semantic reasoning [20, 109]. The original approach [84] adopts a learning-to-rank framework (RankSVM) [49] that learns a global linear ranking function for each attribute. The model uses pairwise supervision—pairs of images ordered according to their perceived attribute strength based on human annotators—and trains a ranking function that preserves those orderings. Given a novel pair of images, the ranker indicates which image has the attribute more. Subsequently, non-linear ranking functions [69], combining feature-specific rankers [18], and multi-task learning [17] have all shown to further improve accuracy. More recently, the success of deep networks has motivated end-to-end frameworks for learning features and attribute ranking functions simultaneously [77, 107, 108, 122]. While these deep models offer a higher learning capacity compared to their “shallow” counterparts, they also come with a greater need for labeled data and higher computation cost.

Aside from these learning-to-rank formulations, researchers have applied the Elo rating system for biometrics [94], and regression over “cumulative attributes” for age and crowd density estimation [16]. Other work investigates features tailored for attribute comparisons, such as facial landmark detectors [96] and “visual chains” to discover

relevant parts [117].

While all prior methods produce a single global function for each attribute, my work in Chapter 4 proposes to learn local functions tailored to each comparison at hand. Furthermore, at test time, we move beyond the standard relative decisions (e.g., image A is *more sporty* than image B) by permitting a third equality option based on distinguishability, something that no existing work has attempted. Meanwhile, our dense supervision approach in Chapter 6 contributes to the learning by densifying the label space, thus boosting the overall performance of all rankers regardless of the learning algorithm used. Our experiments support this claim for two popular ranking frameworks—the above mentioned RankSVM [84] and a Siamese deep convolutional neural network (CNN) [107].

2.2 Fine-Grained Visual Tasks

The fact that humans exhibit inconsistencies in their comparisons is well known in social choice theory and preference learning [10]. In existing global models [18, 27, 59, 69, 75, 84, 96], intransitive constraints would be unaccounted for and treated as noise. We are interested in modeling attributes where there *is* consensus about comparisons, only they are subtle, or in other words, *fine-grained*. Rather than personalize a model towards an observer [2, 14, 57], we want to discover the (implicit) map of where the consensus for decision boundaries in attributes exists. The attribute calibration method of [97] post-processes attribute classifier outputs so they can be fused for multi-attribute search. Our local methods are also conscious that differences in attribute outputs taken at “face value” can be misleading, but our goal and approach are entirely different.

In the facial attractiveness ranking method of [14], the authors train a hierarchy of support vector machine (SVM) classifiers to recursively push an image into buckets of more/less attractive faces. The leaf nodes contain images “unrankable” by the human subject, which can be seen as indistinguishability for the specific attribute of human attractiveness. Nonetheless, the proposed method is not applicable to our problem. It learns a ranking model specific to a single human subject, whereas we learn a subject-independent model. Furthermore, the training procedure [14] has limited scalability, since the subject must rank *all* training images into a partial order. In our domains of interest, where thousands or more training instances are standard, getting a reliable global partial order on all images remains an open challenge.

Research on fine-grained visual *categorization* aims to recognize objects in a single domain, e.g., birds [11, 29], planes [74], and cars [121]. While such problems also require making distinctions among visually close instances, our goal is to compare attributes, not categorize objects.

2.3 Local Learning

Existing local learning algorithms primarily vary in how they exploit the labeled instances nearest to a test point. One strategy is to identify a fixed number of neighbors most similar to the test point, then train a model with only those examples (e.g., linear regression [4], neural network [9], SVM [132]). Alternatively, the nearest training points can be used to learn a transformation of the feature space (e.g., Linear Discriminant Analysis); after projecting the data into the new space, the model is better tailored to the query’s neighborhood properties [23, 26, 38, 115]. In *local selection* methods, strictly the subset of nearby data is used, whereas in *locally weighted* methods, all training points are used but weighted according to their distance [4]. For all these prior methods, a test case is a new data point and its neighboring examples are identified by nearest neighbor search. In contrast, my approach in Chapter 4 learns local ranking functions for comparisons, which requires identifying analogous neighbor *pairs* in the training data. Furthermore, we also explore how to *predict* the variable-size set of training instances that will produce an effective discriminative model for a given test instance.

In information retrieval, local learning methods have been developed to sort documents by their relevance to query keywords [5, 26, 34, 71]. They take strategies quite similar to the above, e.g., building a local model for each cluster in the training data [71], projecting training data onto a subspace determined by the test data distribution [26], or building a model with only the query’s neighbors [5, 34]. Though a form of ranking, the problem setting in all these methods is quite different from ours. There, the training examples consist of queries and their respective sets of ground truth “relevant” and “irrelevant” documents, and the goal is to learn a function to rank a keyword query’s relevant documents higher than its irrelevant ones. In contrast, we have training data comprised of paired comparisons, and the goal is to learn a function to compare a novel query pair.

2.4 Attribute and Image Synthesis

The key to creating dense supervision from synthetic images is a generative model that can progressively modify the target attribute while preserving the rest (my goal in Chapter 6 and 7). Attribute-specific alterations have been considered in several recent methods, primarily for face images. Some target a specific domain and attribute, such as methods to enhance the “memorability” [55] or age [53] of facial photos, to edit outdoor scenes with transient attributes like weather [65], or to modify 3D attributes such as pose and depth of objects [22].

Alternatively, image synthesis is of interest in the low-shot learning community, where training data for the novel classes are severely limited [22, 37, 64, 78]. The primary draw is the ability to “hallucinate” variability for learning where the variability is practically non-existent. While synthesis often happens in the image space, it can also happen in the feature space directly [22, 37, 40, 64]. Though similarly motivated, our focus is not restricted to low-shot cases.

Meanwhile, the success of deep neural networks for image generation (i.e., Generative Adversarial Nets (GAN) [35, 43, 45, 76, 92, 131, 133, 137] or Variational Auto-Encoder (VAE) [36, 56, 61]) opens the door to learning how to generate images conditioned on desired properties [24, 68, 83, 118, 119]. For example, a conditional multimodal auto-encoder can generate faces from attribute descriptions [83], and focus on identity-preserving changes [68]. Furthermore, conditional models can also synthesize an image based on an input, either a label map [45, 137] or a latent attribute label [67, 76, 111, 118]. To densify the pairwise label space for fine-grained comparisons, we employ the state-of-the-art model of [118] due to its generality. We show how to sample images using this generator to “fill in the gaps” in the label space. Whereas the above methods aim to produce an image for human inspection, we aim to generate dense supervision for learning algorithms. We are the first to propose direct image generation as a solution to the sparsity of supervision problem.

Lastly, despite the success of deep neural networks, research has demonstrated their sensitivity to small perturbations to the input image through “fooling networks” [79, 80]. This field of work relates to our active image generation approach in Chapter 7, where rather than using adversarial generation to understand how features influence a classifier, our goal is to synthesize the very training samples that (once labeled by human

annotators) will strengthen a learned ranker. Unlike any of the above, we create images for active query synthesis.

2.5 Learning using Synthetic Images

The use of synthetic images as training data—as we have proposed for learning ranking models—has been explored to a limited extent, primarily for human bodies. Taking advantage of high quality graphics models for humanoids, rendered images of people from various viewpoints and body poses provide free data to train pose estimators [100, 101, 112] or person detectors [90]. Using the first frame of a video as reference, one can personalize a pose estimator by synthesizing deformations [85] or train an object tracker by synthesizing plausible future frames [54]. For objects beyond people, recent work considers how to exploit non-photorealistic images generated from 3D CAD models to augment training sets for object detection [88, 120] and for indoor scene understanding [134], or words rendered in different fonts for text recognition [46]. In addition, others explore how to adversarially generate “hard” low-level transformations that are label-preserving for body pose estimation [89], greedily select useful transformations for image classification [87], or actively evolve part-based 3D shapes to learn shape from shading [120]. Concurrent work [103] comes closest in motivation to our dense supervision approach by recognizing the potential to use synthetic images for learning. However, unlike [103], we use the generated synthetic images directly for training, whereas [103] first learns a refiner network to transform the synthetic images into more realistic images.

While these methods share our concern about the sparsity of supervision, our focus on attributes and ranking is unique. Furthermore, most methods assume a graphics engine and 3D model to render new views with desired parameters (pose, viewpoint, etc.). In contrast, we investigate (in Chap. 6 and 7) images generated from a 2D image synthesis engine in which the modes of variation are controlled by a *learned* model. Being data-driven can offer greater flexibility, allowing tasks beyond those requiring a 3D model, and variability beyond camera pose and lighting parameters, albeit in exchange for noisier generation.

As discussed above, in the low-shot recognition regime, several recent methods explore creative ways to hallucinate the variability around sparse real examples [22, 37, 39, 64, 98], typically leveraging the observed inter-sample transformations to guide synthesis in feature-space. However, whereas most of these methods use manually defined

heuristics to sample synthetic images, we show in Chapter 7 how to dynamically derive the images most valuable to training, via an adversarial control module learned jointly with the attribute ranker.

2.6 Active Learning

Active learning has been studied for decades [99]. For visual recognition problems, *pool-based* methods are the norm: the learner scans a pool of unlabeled samples and iteratively queries for the label on one or more of them based on uncertainty or expected model influence (e.g., [32, 113, 114, 135]). Active ranking models adapt the concepts of pool-based learning to select pairs for comparison [70, 91]. Hard negative mining—often used for training object detectors [30, 102]—also focuses the learner’s attention on useful samples, though in this case from a pool of already-labeled data. Rather than display one query image to an annotator, the approach in [44] selects a sample from the pool then displays a synthesized image *spectrum* around it in order to elicit feature points likely to be near the true linear decision boundary for image classification. We do not perform pool-based active selection. Unlike any of the above, our approach in Chapter 7 *creates* image samples that (once labeled) should best help the learner, and it does so in tight coordination with the ranking algorithm.

In contrast to pool-based active learning, active *query synthesis* methods request labels on novel samples from a given distribution [1, 3, 99, 136]. When the labeler is a person (as opposed to an oracle or experimental outcome), a well known challenge with query synthesis is that the query may turn out to be unanswerable [6]. Perhaps accordingly, there is very limited prior work attempting active query synthesis for image problems, and to our knowledge they are limited to toy cases like MNIST digits [136]. Our active image generation work in Chapter 7 capitalizes on the recent advances in image generation discussed above to create photorealistic queries that are most often answerable. Furthermore, rather than sample from an input distribution, our selection approach is discriminative: it optimizes the latent parameters of an image pair that directly affect the current deep ranking model.

Chapter 3

Datasets

In this chapter, I present the datasets across various domains that are used throughout this thesis. The supervision on these datasets comes in the form of *relative* annotations between a pair of images, i.e., given a target attribute \mathcal{A} , image i is perceived to have “more/less” (or “equal”) of \mathcal{A} than image y . Due to the fine-grained nature of the comparison tasks, we require the pairwise supervision to have the utmost precision, something that most of the existing datasets do not enforce.

First, we present our newly collected large-scale shoe dataset (Sec. 3.1), followed by our subsequent improvements with an additional lexicon and supervision (Sec. 3.2). Then, we present the various face datasets (Sec. 3.3) and the scene dataset (Sec. 3.4) used in our experiments. For our methods in Chapters 6 and 7, we also make use of synthetically generated image pairs for training ranking models. However, we will leave the specific details of the synthetic image datasets to the respective chapters.

3.1 UT-Zap50K: Fine-Grained Shoe Dataset

To facilitate the evaluation of our fine-grained models, we collected a new UT Zappos50K dataset (**UT-Zap50K**) specifically targeting the fine-grained attribute comparison task.¹ The dataset is fine-grained due to two factors: 1) it focuses on a narrow domain of content, and 2) we develop a two-stage annotation procedure to isolate those comparisons that humans find perceptually very close.

The image collection is created in the context of an online shopping task, with 50,000 catalog shoe images from Zappos.com. For online shopping, users care about precise visual differences between items. For instance, it is more likely that a shopper is deciding between two pairs of similar men’s running shoes instead of between a woman’s high heel and a man’s slipper. The images are roughly 150×100 pixels and shoes are

¹UT-Zap50K and all related data are available for download at vision.cs.utexas.edu/projects/finegrained



Figure 3.1: Sample images from each of the high-level shoe categories of UT-Zap50K.

pictured in the same orientation for convenient analysis (Fig. 3.1). For each image, we also collect its meta-data (shoe type, materials, manufacturer, gender, etc.) that are used to filter the shoes on Zappos.com (Tab. 3.1).

Using Mechanical Turk (mTurk), we collect ground truth comparisons for four relative attributes: *open*, *pointy at the toe*, *sporty*, and *comfortable*. The attributes are manually selected for their potential to exhibit fine-grained differences. A worker is shown two images and an attribute name, and must make a relative decision (more, less, equal) and report the confidence of his decision (high, mid, low). We repeat the same comparison for five workers in order to vote on the final ground truth. We collect 12,000 total pairs, 3,000 per attribute. After removing the low confidence (less than “mid” confidence), the low agreement (less than 80% worker agreement), and “equal” pairs, each attribute has between 1,500 to 1,800 total ordered pairs.

Of all the possible 50,000² pairs we could get annotated, we want to prioritize the fine-grained pairs. To this end, first, we sampled pairs with a strong bias (80%) towards intra-category and -gender images (based on the meta-data). We call this collection **UT-Zap50K-1**. We found $\sim 40\%$ of the pairs came back labeled as “equal” for each attribute. While the “equal” label can indicate that there is no perceivable difference in the attribute, we also suspected that it was an easy fallback response for cases that required a little more thought—that is, those showing fine-grained differences. Thus, we next posted the pairs rated as “equal” (4,612 of them) back onto mTurk as new tasks, but *without* the “equal” option. We asked the workers to look closely, pick one image over the other, and give a one sentence rationale for their decisions. We call this set **UT-Zap50K-2**. Screenshots of the mTurk tasks are shown in Figure 3.2.

Interestingly, the workers are quite consistent on these pairs, despite their difficulty. Overall, 63% of the fine-grained pairs (and 66% of the coarser pairs) had at least four out of five workers agree on the same answer with above average confidence. This

Table 3.1: Taxonomy of potential meta-data labels for each image in the UT-Zap50K dataset. Each image is not guaranteed to have all labels, except for “Category” and “SubCategory”, but could have multiple values within each label.

Labels	Potential Values
Category	{shoes, boots, sandals, slippers}
SubCategory	{ankle, athletic, flats, heels, loaders, ...}
HeelHeight	{under 1in, 1in-1 3/4in, 2in-2 3/4in, ...}
Insole	{leather, padded, textile, memory foam, polyurethane, ...}
Closure	{ankle strap, buckle, pull-on, lace up, zipper, ...}
Gender	{men, women, boys, girls}
Material	{leather, rubber, suede, neoprene, sheepskin, ...}
ToeStyle	{open, round, capped, snub, peep, ...}


consistency ensures we have a dataset that is both fine-grained as well as reliably ground truthed. Compared to an existing Shoes attribute dataset [7] with relative attributes [59], UT-Zap50K is about $3.5\times$ larger, offers meta-data and $10\times$ more comparative labels, and most importantly, specifically targets fine-grained tasks.

Finally, while our pre-processing methodology of the worker annotations above is a standard process, my equality prediction work in Chapter 5 requires even more precision in the annotations. As such, we create another set of fine-grained shoe labels, which we call **UT-Zap50K-EQ**, specifically for this work with an even more stringent pre-process methodology. We require the use of both ordered pairs \mathcal{P}_o and “equal” pairs \mathcal{P}_e in this case. For \mathcal{P}_o , we use all coarse and fine-grained pairs for which all five workers agreed and had high confidence. Even though the fine-grained pairs might be visually similar, if all five workers could come to agreement with high confidence, then the images are most likely distinguishable. For \mathcal{P}_e , we use all fine-grained pairs with three or four workers in agreement and only “mid” confidence. Since the fine-grained pairs have already been presented to the workers twice, if they are still unable to come to an consensus with high confidence, then the images are most likely indistinguishable. The resulting dataset has 4,778 total annotated pairs, consisting of on average 800 ordered and 350 indistinguishable pairs per attribute.

3.2 Improved UT-Zap50K with a Fine-Grained Lexicon

Given the initial success of our UT-Zap50K dataset for fine-grained comparison tasks, which will be described in Chapters 4 and 5, we next expanded it by turning our at-

Image Pair #7



is more equally less **"open"** than high mid low


more equally less **"pointy at the toe"** high mid low

more equally less **"sporty"** with high mid low **confidence.**

more equally less **"comfortable"** high mid low

(a) Round one to obtain UT-Zap50K-1.

Image Pair #4 (comfortable)



Shoe A **Shoe B**

Shoe A is more "comfortable" than **Shoe B**.

Shoe B is more "comfortable" than **Shoe A**.

I'm **very confident** about my decision.

I'm **somewhat confident** about my decision.

I'm **not confident** about my decision.

Briefly explain your reasoning for this choice.

(b) Round two to obtain UT-Zap50K-2.

Figure 3.2: Sample questions from mTurk data collection framework.

tention towards the selection of the attribute names. The *lexicon* of attributes used in existing relative attributes datasets, including our initial UT-Zap50K, is selected based on intuitions, i.e., words that seem domain relevant [60] or words that seem to exhibit the most subtle fine-grained differences.

However, this kind of intuition is inherently biased by the person making the selection. Therefore, to address this limitation, we (1) use crowdsourcing to mine for an attribute lexicon that is explicitly fine-grained, and (2) collect a large number of pairwise labels for each attribute in that lexicon.

Given a pair of images, we ask mTurk workers to complete the sentence, “Shoe A is a *little more* ⟨insert word⟩ than Shoe B” using a single word. They are instructed to identify subtle differences between the images and provide a short rationale. The



Figure 3.4: Sample image pairs from the improved UT-Zap50K dataset for five attributes from the fine-grained lexicon. The top section represents ordered pairs (the left image has “more” of the attribute than the right image) while the bottom section represents “equal” pairs. Note that some images can look drastically different overall while still exhibiting subtle differences in the target attribute space, and vice-versa.

3.3 Face Datasets

Aside from our UT-Zap50K dataset in the shoe domain, the most popular category of datasets used for fine-grained comparison is faces. Out of those datasets, the LFW-10 dataset [96] is the only one that contains pairwise supervision at the instance-level. The LFW-10 dataset consists of 2,000 face images taken from Labeled Faces in the Wild (LFW) [42]. It contains 10 attributes: *bald*, *dark hair*, *big eyes*, *good looking*, *masculine*, *mouth open*, *smiling*, *visible teeth*, *visible forehead*, and *young*. After pruning pairs with less than 80% agreement from the workers, there are 600 pairwise labels on average per attribute. Figure 3.5 shows sample face image pairs from the dataset.

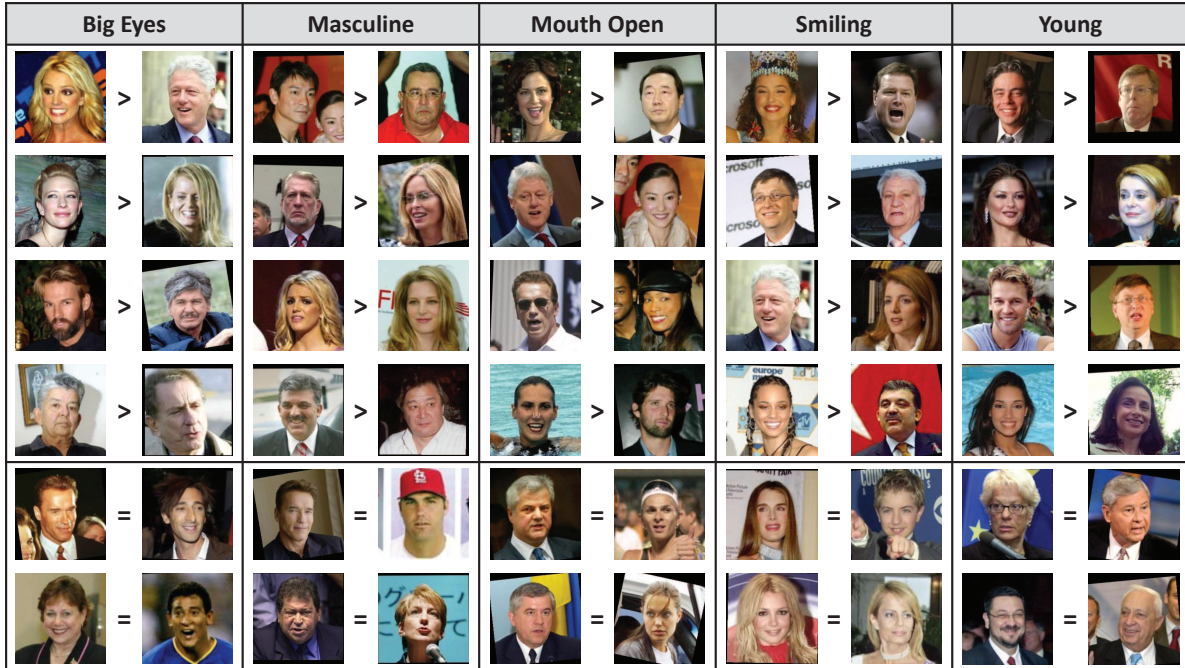


Figure 3.5: Sample image pairs from the LFW-10 dataset [96], in the same format as Figure 3.4.

Another relevant face dataset is the Public Figures [63] (**PubFig**), which is one of the pioneer datasets used for relative comparison tasks (Fig. 3.6, right) and has been used as the benchmark for many prior works. PubFig contains 772 images with 11 attributes: *male*, *white*, *young*, *smiling*, *chubby*, *forehead*, *bushy eyebrows*, *narrow eyes*, *pointy nose*, *big lips*, and *round face*. However, instead of instance-level supervision, PubFig contains only category-level supervision (e.g., “Viggo *smiles* less than Miley”) that is propagated down uniformly to all image instances [59, 69, 84]. As such, there are on average over 20,000 pairwise labels per attribute.

In addition, for our experiments in Chapter 6, we form a small face dataset called **PFSmile** using images from the Public Figures dataset (PubFig) [63, 84]. We select eight frontal images each from eight random individuals, with the frontal images showing different degrees of *smilingness* for the given individual (e.g., images of Zach Efron going from not smiling at all to fully smiling). We use smiling because it is the only PubFig attribute that manifests fine-grained changes on the same individual (e.g., it does not display Zach both as *bald* and *less bald*). This limitation of the data actually reinforces the difficulty of manually curating images with subtle differences for learning, as we will see in Chapter 6. We collect labels on all possible pairwise comparisons among images of the same individual. After pruning, there are 211 pairwise labels for PFSmile.

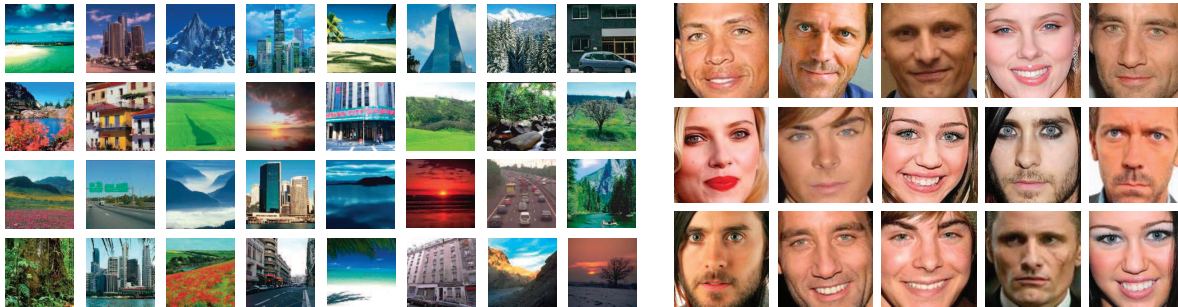


Figure 3.6: Sample images from the Outdoor Scene Recognition dataset [82] (left) and the Public Figure Faces dataset [63] (right).

Finally, with the same motivation as UT-Zap50K-EQ above, we create another set of fine-grained face labels, which we call **LFW-10-EQ**. For \mathcal{P}_o , we use all pairs labeled “more” or “less” by at least four workers. For \mathcal{P}_e , we use pairs where at least four workers said “equal”, as well as pairs with the same number of “more” and “less” votes. The latter reflects that a split in decision signals indistinguishability. Due to the smaller scale of LFW-10, we could not enforce full agreement on ordered pairs as we did for UT-Zap50K-EQ; such pruning would eliminate most of the labeled data. The resulting dataset has 5,543 total annotated pairs, on average 230 ordered and 320 indistinguishable pairs per attribute.

3.4 Scene Dataset

Finally, the Outdoor Scene Recognition [82] (**OSR**) dataset is another one of the pioneer datasets used for relative comparison tasks (Fig. 3.6, left). OSR contains 2,688 scene images with 6 attributes: *natural*, *open*, *perspective*, *large size*, *diagonal*, and *close depth*. Just like PubFig, OSR contains only category-level supervision that is propagated down uniformly to all image instances [59, 69, 84], resulting in over 20,000 pairwise labels per attribute on average.

To our knowledge, our UT-Zap50K and LFW-10 are the only existing datasets that contain *image-level* comparisons (e.g., “this particular shoe is *more pointy* than that particular shoe”). These image-level comparisons are more costly to obtain but essential for testing fine-grained attributes thoroughly.

Having defined all the datasets that will support the experiments in this thesis, we now present the proposed approaches.

Chapter 4

Local Ranking Functions for Attributes

When performing attribute comparisons, existing methods train a global ranking function using all available constraints, with the implicit assumption that more training data should only help better learn the target concept. While such an approach tends to capture the coarse visual comparisons, it can be difficult to derive a single set of model parameters that adequately represents both these big-picture contrasts *and* more subtle fine-grained comparisons (recall Fig. 1.1).¹ For example, for a dataset of shoes, it will map all the sneakers on one end of the *formal* spectrum, and all the high heels on the other, but the ordering among closely related high heels will not show a clear pattern. This suggests there is an interplay between the model capacity and the density of available training examples, prompting us to explore local learning solutions.

In this chapter, we propose a novel local learning approach to learn a custom attribute-specific model on-the-fly for each comparison pair at hand. We first present a brief overview of two types of widely used ranking functions (Sec. 4.1.1 and 4.1.2). Next, we introduce our local ranking approach (Sec. 4.2.1) and the mechanism for selecting fine-grained neighboring pairs with attribute-specific metric learning (Sec. 4.2.2). On three challenging datasets from distinct domains, including our newly curated large shoe dataset UT-Zap50K that focuses on fine-grained attribute comparisons (Sec. 3.1), we show our approach improves the state-of-the-art in relative attribute predictions (Sec. 4.3). After the results, we present an extension of the local attribute learning idea that learns the *neighborhood* of relevant training data that ought to be used to train a model on-the-fly (Sec. 4.4).

The work in this chapter was published in CVPR 2014 [125], NIPS 2014 [126] and a book chapter [124].

¹This is true particularly for common learning algorithms like support vector machines, though potentially less problematic in very high capacity learning algorithms with deep networks. We explore both in this thesis.

4.1 Ranking Functions for Relative Attributes

Relative Attributes, as originally proposed by Parikh and Grauman [84], treats the attribute comparison task as a learning-to-rank problem. The idea is to use ordered pairs (and optionally “equal” pairs) of training images to train a ranking function that will generalize to new images. Compared to learning a regression function, the ranking framework has the advantage that training instances are themselves expressed comparatively, as opposed to requiring a rating of the absolute strength of the attribute per training image.

For each attribute \mathcal{A} (e.g., *comfortable*) to be learned, we take as input two sets of annotated training image pairs. The first set consists of ordered pairs, $\mathcal{P}_o = \{(i, j)\}$, for which humans perceive image i to have the attribute more than image j . That is, each pair in \mathcal{P}_o has a “noticeable difference”. The second set consists of unordered, or “equal” pairs, $\mathcal{P}_e = \{(m, n)\}$, for which humans cannot perceive a difference in attribute strength. Refer to Section 3.1 for discussion on how such human-annotated data can be reliably collected.

Let $\mathbf{x}_i \in \mathbb{R}^d$ denote the d -dimensional feature descriptor for image i , such as a GIST descriptor [110], a color histogram, or the vectorized image pixels, and let $R_{\mathcal{A}} : \mathbb{R}^d \rightarrow \mathbb{R}$ be a ranking function for attribute \mathcal{A} . In my thesis work, I explore two main families of ranking functions: large-margin rankers and deep rankers. Both are widely used in the literature [8, 60, 84, 107, 108, 122] and offer different trade-offs, to be discussed below.

4.1.1 Large-Margin Ranking Functions

Using a large-margin approach based on the SVM-Rank framework [49], the goal for a global relative attribute is to learn the parameters $\mathbf{w}_{\mathcal{A}} \in \mathbb{R}^d$ that optimize the rank function, $R_{\mathcal{A}}(\mathbf{x}) = \mathbf{w}_{\mathcal{A}}^T \mathbf{x}$. These parameters aim to preserve the orderings in \mathcal{P}_o , maintaining a margin between them in the 1D output space, while also minimizing the separation between the unordered pairs in \mathcal{P}_e . By itself, the problem is NP-hard, but [49] introduces slack variables and a large-margin regularizer to approximately solve it. The learning objective is:

$$\begin{aligned}
\text{minimize} \quad & \left(\frac{1}{2}\|\mathbf{w}_{\mathcal{A}}\|_2^2 + C \left(\sum \xi_{ij}^2 + \sum \gamma_{mn}^2\right)\right) & (4.1) \\
\text{s.t.} \quad & \mathbf{w}_{\mathcal{A}}^T(\mathbf{x}_i - \mathbf{x}_j) \geq 1 - \xi_{ij}; \forall (i, j) \in \mathcal{P}_o \\
& |\mathbf{w}_{\mathcal{A}}^T(\mathbf{x}_m - \mathbf{x}_n)| \leq \gamma_{mn}; \forall (m, n) \in \mathcal{P}_e \\
& \xi_{ij} \geq 0; \gamma_{mn} \geq 0,
\end{aligned}$$

where the constant C balances the regularizer and ordering constraints, and ξ_{ij} and γ_{mn} denote slack variables. By projecting images onto the resulting hyperplane $\mathbf{w}_{\mathcal{A}}$, we obtain a 1D global ranking for that attribute, e.g., from least to most *comfortable*.

Given a test pair $(\mathbf{x}_r, \mathbf{x}_s)$, if $R_{\mathcal{A}}(\mathbf{x}_r) > R_{\mathcal{A}}(\mathbf{x}_s)$, then image r exhibits the attribute more than image s , and vice versa. While [84] uses this linear formulation, it is also kernelizable and so can produce non-linear ranking functions. A large-margin ranker’s ease of training (convex optimization) and its ability to handle pairwise inconsistencies make it suitable for our purpose. Furthermore, the SVM-Rank approach can be successfully trained with relatively few labeled instances, a scenario that is practically important for attribute comparisons tasks, where pairwise supervision is often limited (see Sec. 3.1).

Our local approach defined in Section 4.2 draws on this particular large-margin formulation (referred to as RankSVM from here on). As we will see later in Chapters 6 and 7, while the recent deep neural networks (defined next in Sec. 4.1.2) have higher learning capacities (even than with a non-linear kernel), their need for a large amount of training data makes them less suitable for our local approach.

4.1.2 Deep Ranking Functions

With the recent success of deep neural networks, some research has extended relative attributes into the deep domain by designing end-to-end networks that simultaneously learn the image features and the ranker [107, 108, 122]. One common aspect of these end-to-end networks is the use of the RankNet algorithm [12], which we will briefly overview below.

RankNet is a neural network based ranking algorithm with a probabilistic cost function. Given image i and its feature descriptor \mathbf{x}_i , we would like to learn a ranking function $R_{\mathcal{A}} : \mathbb{R}^d \rightarrow \mathbb{R}$, consisting of a series of neural network modules, that outputs

the corresponding real-valued strength v_i for attribute \mathcal{A} . Let $(\mathbf{x}_i, \mathbf{x}_j)$ be a pair of image features (from either \mathcal{P}_o or \mathcal{P}_e) and t_{ij} be its target probability indicating the probability of $R_{\mathcal{A}}(\mathbf{x}_i)$ being higher valued than $R_{\mathcal{A}}(\mathbf{x}_j)$. RankNet maps this rank estimate to a pairwise posterior probability p_{ij} using a logistic function:

$$p_{ij} = \frac{1}{1 + e^{-(v_i - v_j)}}. \quad (4.2)$$

The ranking loss is then defined as:

$$\mathcal{L}_{rank} = -t_{ij} \log(p_{ij}) - (1 - t_{ij}) \log(1 - p_{ij}), \quad (4.3)$$

which is a standard cross-entropy loss where $t_{ij} = 1$ if $(i, j) \in \mathcal{P}_o$ or $t_{ij} = 0.5$ if $(i, j) \in \mathcal{P}_e$. This loss function is ideal for ranking purposes as it asymptotes to a linear function, making it more robust to noise compared to a regular quadratic function. Furthermore, it can also handle equality cases where the cost becomes symmetric. The overall function can be trained using stochastic gradient descent or its variations.

Compared to the large-margin ranker above, a deep ranker has a much higher learning capacity. Using the RankNet algorithm, we are also free to design the learning network without restrictions, as long as it outputs a v_i at the end. However, the higher learning capacity also comes with a greater need for labeled data and a higher computation cost. Existing methods rely on data augmentation techniques in the image-space (mirroring, scaling, etc.) to compensate for the data needs, something that we discuss in detail in Chapter 6 and 7. In the same chapters, we also employ the deep rankers in our dense supervision framework.

4.2 Approach

With the understanding of the existing ranker choices, in the approach presented in this chapter, I focus on improving fine-grained performance using RankSVM, the large-margin ranker. A key premise of local learning is that improving accuracy is not simply a matter of using a higher capacity learning algorithm. While a low capacity model can perform poorly in well-sampled areas, unable to sufficiently exploit the dense training data, a high capacity model can produce unreliable (yet highly confident) decisions in poorly sampled areas of the feature space [9]. Different properties are required in different

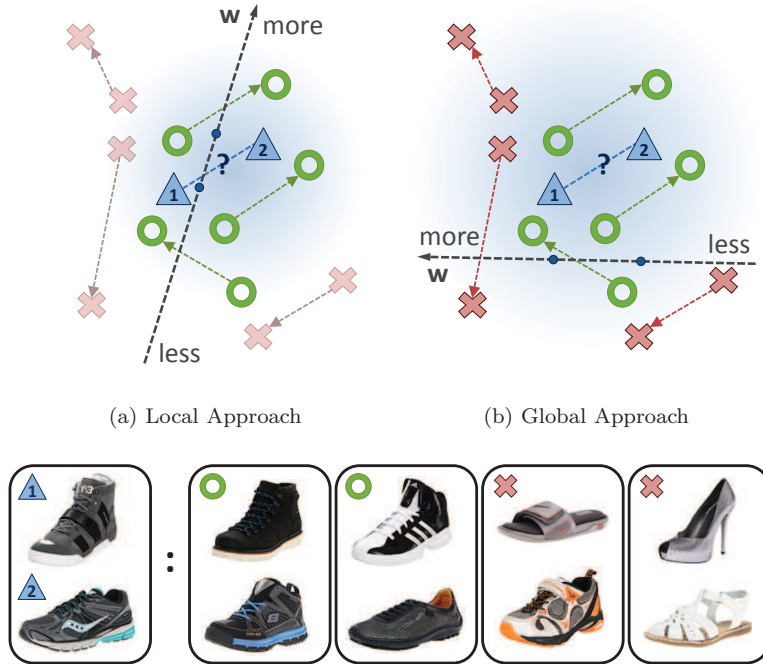


Figure 4.1: Given a novel test pair (blue \triangle) in a learned metric space, my proposed local approach (a) selects only the most relevant neighbors (green \circ) for training, which leads to ranking test image 2 over 1 in terms of *sporty*. In contrast, the standard global approach (b) defined in Section 4.1.1 uses all training data (green \circ & red \times) for training; the unrelated training pairs dilute the training data. As a result, the global model accounts largely for the coarse differences, and incorrectly ranks test image 1 over 2. The end of each arrow points to the image with *more* of the attribute (*sporty*). Note that the rank of each point is determined by its *projection* onto w .

areas of the feature space. Furthermore, in our visual ranking domain, we can expect that as the amount of available training data increases, more human subjectiveness and ordering inconsistencies will emerge, further straining the validity of a single global function.

4.2.1 Local Learning for Visual Comparisons

Our idea is to explore a local learning approach for attribute ranking. The idea is to train a ranking function tailored to each novel pair of images $X_q = (\mathbf{x}_r, \mathbf{x}_s)$ that we wish to compare. We train the custom function using only a subset of all labeled training pairs, exploiting the data statistics in the neighborhood of the test pair. Let $\mathcal{P}_A = \{\mathcal{P}_o \cup \mathcal{P}_e\}$ be all training pairs with respect to attribute \mathcal{A} . In particular, we sort \mathcal{P}_A by its similarity to $(\mathbf{x}_r, \mathbf{x}_s)$, then compose a local training set \mathcal{P}'_A consisting of the top K neighboring pairs, $\mathcal{P}'_A = \{(\mathbf{x}_{k1}, \mathbf{x}_{k2})\}_{k=1}^K$. We explain in the next section how we define similarity between pairs. Then, we train a ranking function on-the-fly, and apply it to compare



Figure 4.2: Illustration of *analogous* image pairs. Comparing a novel test pair to three training pairs, with each column representing a pair of images. Left: Both top images and bottom images are visually similar, resulting in the lowest pairwise distance. Middle: While the top images are dissimilar, the bottom images are similar, keeping the pairwise distance relatively low. Right: Both top images and bottom images are visually dissimilar, resulting in the highest pairwise distance. Pair #1 is the most *analogous* pair to the novel test pair in this case.

the test images. Experiments here use the RankSVM objective (Eq. 4.1). Thus, while the capacity of the trained models will be fixed throughout the feature space, crucially, the composition of their training sets and the resulting models will vary.

While simple, our framework directly addresses flaws that hinder existing methods. By restricting training pairs to those visually similar to the test pair, the learner can zero in on features most important for that kind of comparison. Such a fine-grained approach helps to eliminate ordering constraints that are irrelevant to the test pair. For instance, when evaluating whether a high-topped athletic shoe is more or less *sporty* than a similar looking low-topped one, our method will exploit pairs with similar visual differences, as opposed to trying to accommodate in a single global function the contrasting sportiness of sneakers, high heels, and sandals (Fig. 4.1).

4.2.2 Selecting Fine-Grained Neighboring Pairs

A key factor to the success of the local rank learning approach is how we judge similarity between pairs. Intuitively, we would like to gather training pairs that are somehow *analogous* to the test pair, so that the ranker focuses on the fine-grained visual differences that dictate their comparison. This means that not only should individual members of the pairs have visual similarity, but also the visual contrasts between the two test pair images should mimic the visual contrasts between the two training pair images (Fig. 4.2). In addition, we must account for the fact that we seek comparisons along a particular attribute, which means only certain aspects of the image appearance are relevant.

To fulfill these desiderata, we define a paired distance function that incorporates

attribute-specific metric learning. Let $X_q = (\mathbf{x}_r, \mathbf{x}_s)$ be the test pair, and let $X_t = (\mathbf{x}_u, \mathbf{x}_v)$ be a labeled training pair for which $(u, v) \in \mathcal{P}_A$. We define their distance as:

$$D_A(X_q, X_t) = \min \left(D'_A((\mathbf{x}_r, \mathbf{x}_s), (\mathbf{x}_u, \mathbf{x}_v)), D'_A((\mathbf{x}_r, \mathbf{x}_s), (\mathbf{x}_v, \mathbf{x}_u)) \right), \quad (4.4)$$

where D'_A is the product of the two items' distances:

$$D'_A((\mathbf{x}_r, \mathbf{x}_s), (\mathbf{x}_u, \mathbf{x}_v)) = d_A(\mathbf{x}_r, \mathbf{x}_u) \times d_A(\mathbf{x}_s, \mathbf{x}_v). \quad (4.5)$$

The product reflects that we are looking for pairs where each image is visually similar to one of those in the novel pair. It also ensures that the constraint pairs are evaluated for distance as a pair instead of as individual images. The minimum in Equation 4.4 and the swapping of $(\mathbf{x}_u, \mathbf{x}_v) \rightarrow (\mathbf{x}_v, \mathbf{x}_u)$ in the second term ensure that we account for the unknown ordering of the test pair. When learning a local ranking function for attribute \mathcal{A} , we sort neighbor pairs for X_q according to D_A , then take the top K to form \mathcal{P}'_A .

When identifying neighbor pairs, rather than judge image distance d_A by the usual Euclidean distance on global descriptors, we want to specialize the function to the particular attribute at hand. That is because often a visual attribute does not rely equally on each dimension of the feature space, whether due to the features' locations or modality. For example, if judging image distance for the attribute *smiling*, the localized region by the mouth is likely most important. For fine-grained comparisons, global similarity alone is insufficient. We need to focus on those that are similar in terms of the property of interest.

To this end, we learn a Mahalanobis metric:

$$d_A(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M}_A (\mathbf{x}_i - \mathbf{x}_j), \quad (4.6)$$

parameterized by the $d \times d$ positive definite matrix \mathbf{M}_A . We employ the information-theoretic metric learning (ITML) algorithm [19], due to its efficiency and kernelizability. Given an initial $d \times d$ matrix \mathbf{M}_{A_0} specifying any prior knowledge about how the data should be compared, ITML produces the \mathbf{M}_A that minimizes the LogDet divergence

$D_{\ell d}$ from that initial matrix, subject to constraints that similar data points be close and dissimilar points be far:

$$\begin{aligned} \min_{\mathbf{M}_{\mathcal{A}} \succeq 0} \quad & D_{\ell d}(\mathbf{M}_{\mathcal{A}}, \mathbf{M}_{\mathcal{A}_0}) \\ \text{s.t.} \quad & d_{\mathcal{A}}(\mathbf{x}_i, \mathbf{x}_j) \leq c \quad (i, j) \in \mathcal{U}_{\mathcal{A}} \\ & d_{\mathcal{A}}(\mathbf{x}_i, \mathbf{x}_j) \geq \ell \quad (i, j) \in \mathcal{V}_{\mathcal{A}}. \end{aligned} \tag{4.7}$$

The sets $\mathcal{U}_{\mathcal{A}}$ and $\mathcal{V}_{\mathcal{A}}$ consist of pairs of points constrained to be similar and dissimilar, and ℓ and c are large and small values, respectively, determined by the distribution of original distances. We set $\mathbf{M}_{\mathcal{A}_0} = \Sigma^{-1}$, the inverse covariance matrix for the training images. To compose $\mathcal{U}_{\mathcal{A}}$ and $\mathcal{V}_{\mathcal{A}}$, we use image pairs that are human-annotated *according to each attribute* \mathcal{A} . While metric learning is usually used to enhance nearest neighbor classification (e.g., [33, 48]), we employ it to gauge perceived similarity along an attribute.

Figure 4.3 shows example neighbor pairs. They illustrate how our method finds training pairs analogous to the test pair, so the learner can isolate the informative features for that comparison. Note how holistically, the neighbors found with metric learning (FG-LocalPair) may actually look less similar than those found without (LocalPair). However, in terms of the specific attribute, they better isolate the features that are relevant. For example, images of the same exact person need not be most useful to predict the degree of *smiling*, if others better matched to the test pair’s expressions are available (last example). In practice, the local rankers trained with learned neighbors are substantially more accurate.

4.3 Evaluation

To validate our method, we compare it to two state-of-the-art methods as well as informative baselines.

4.3.1 Experimental Setup

Datasets We evaluate on three datasets: our newly curated **UT-Zap50K** shoe dataset (Sec. 3.1) and the two existing **OSR** scene and **PubFig** face datasets (Sec. 3.3 and 3.4). The datasets contain 4, 6, and 11 attributes respectively. We use concatenated GIST [110]


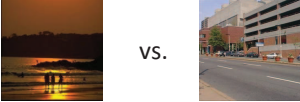
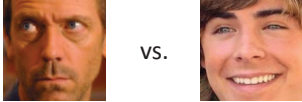






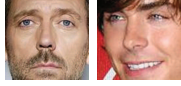
UT-Zap50K (pointy)		OSR (open)		PubFig (smiling)	
					
FG-LocalPair	LocalPair	FG-LocalPair	LocalPair	FG-LocalPair	LocalPair
					
					
					

Figure 4.3: Example fine-grained neighbor pairs for three test pairs (top row) from the datasets tested in this chapter. We display the top 3 pairs per query. FG-LocalPair and LocalPair denote results with and without metric learning (ML), respectively. **UT-Zap50K (pointy)**: ML puts the comparison focus on the tip of the shoe, caring less about the look of the shoe as a whole. **OSR (open)**: ML has less impact, as openness in these scenes relates to their whole texture. **PubFig (smiling)**: ML learns to focus on the mouth/lip region instead of the entire face. For example, while the LocalPair retrieves face pairs that more often contain the same people as the top pair, those instances are nonetheless less relevant for the fine-grained smiling distinction it requires. In contrast, our FG-LocalPair learned metric retrieves nearby pairs that may contain different people, yet are instances where the degree of smiling is most useful as a basis for predicting the relative smiling level in the novel query pair.

and color histogram features for all datasets. For compatibility with prior work, we use the exact same attributes, features, and train/test splits as [69, 84] for OSR and PubFig. Specifically, 240 images from each dataset are designated as the training set, after which the category-level supervision is propagated down to all image instances to form the respective training/testing pairs. Experiments in Chapter 6 and 7 using deep rankers explore learned feature representations.

Setup We run for 10 random train/test splits, setting aside 300 ground truth pairs for testing and the rest for training. We cross-validate C for all experiments, and adopt the same C selected by the global baseline for our approach. We use no “equal” pairs for training or testing rankers. We report accuracy in terms of the percentage of correctly ordered pairs, following [69]. We present results using the same labeled data for all methods.

For learning to rank, our *total* training pairs \mathcal{P}_A consist of only ordered pairs \mathcal{P}_o . For ITML, we use the ordered pairs \mathcal{P}_A for rank training to compose the set of dissimilar pairs \mathcal{V}_A , and the set of “equal” pairs to compose the similar pairs \mathcal{U}_A . We use the default settings for c and ℓ in the authors’ code [19]. The setting of K determines

Table 4.1: Results for the UT-Zap50K dataset. FG-LocalPair represents our proposed method, as we demonstrate the effectiveness of learning from the most analogous neighbors.

	Open	Pointy	Sporty	Comfort		Open	Pointy	Sporty	Comfort
Global [84]	87.77	89.37	91.20	89.93	Global [84]	60.18	59.56	62.70	64.04
RandPair	82.53	83.70	86.30	84.77	RandPair	61.00	53.41	58.26	59.24
LocalPair	88.53	88.87	92.20	90.90	LocalPair	71.64	59.56	61.22	59.75
FG-LocalPair	90.67	90.83	92.67	92.37	FG-LocalPair	74.91	63.74	64.54	62.51

(a) UT-Zap50K-1 with *coarser* pairs.

(b) UT-Zap50K-2 with *fine-grained* pairs.

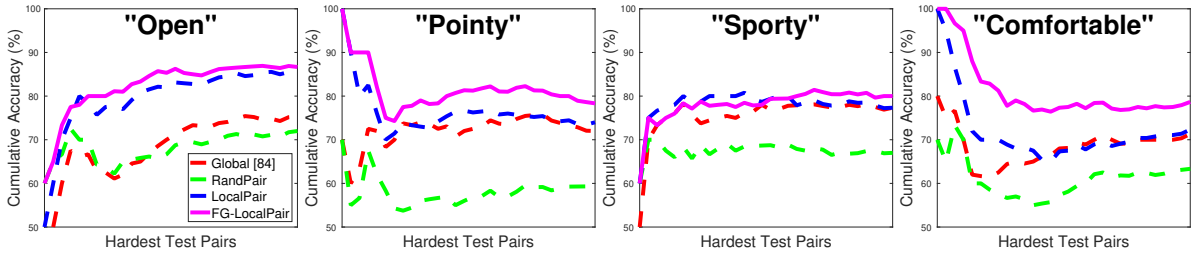


Figure 4.4: Accuracy for the 30 hardest test pairs on UT-Zap50K-1.

“how local” the learner is; its optimal setting depends on the training data and query. As in prior work [9, 132], we simply fix it for all queries at $K = 100$ (though see Sec. 4.4 for a proposed generalization that learns the neighborhood size as well). Values of $K = 50$ to 200 give similar results.

Baselines We compare the following methods:

- **FG-LocalPair**: Our proposed fine-grained approach.
- **LocalPair**: Our approach without the learned metric (i.e., $\mathbf{M}_{\mathcal{A}} = \mathbb{I}$). This baseline isolates the impact of tailoring the search for neighboring pairs to the attribute.
- **RandPair**: A local approach that selects its neighbors randomly. This baseline demonstrates the importance of selecting relevant neighbors.
- **Global**: A global ranker trained with all available labeled pairs, using Equation 4.1. This is the Relative Attributes method [84]. We use the authors’ public code.
- **RelTree**: The non-linear relative attributes approach of [69], which learns a hierarchy of functions, each trained with successively smaller subsets of the data. Code is not available, so we rely on the authors’ reported numbers (available for OSR and PubFig).



Figure 4.5: Example pairs contrasting our predictions to the Global baseline’s. In each pair, the top item is *more sporty* than the bottom item according to ground truth from human annotators. (1) We predict correctly, Global is wrong. We detect subtle changes, while Global relies only on overall shape and color. (2) We predict incorrectly, Global is right. These coarser differences are sufficiently captured by a global model. (3) Both methods predict incorrectly. Such pairs are so fine-grained, they are difficult even for humans to make a firm decision.

4.3.2 Zappos Results

Table 4.1a shows the accuracy on UT-Zap50K-1. Our method outperforms all baselines for all attributes. To isolate the more difficult pairs in UT-Zap50K-1, we sort the test pairs by their intra-pair distance using the learned metric; those that are close will be visually similar for the attribute, and hence more challenging. Figure 4.4 shows the results, plotting cumulative accuracy for the 30 hardest test pairs per split. We see that our method has substantial gains over the baselines (about 20%), demonstrating its strong advantage for detecting subtle differences. Figure 4.5 shows qualitative results.

We proceed to test on even more difficult pairs. Whereas Figure 4.4 focuses on pairs difficult according to the learned metric, next we focus on pairs difficult according to our human annotators. Table 4.1b shows the results for UT-Zap50K-2. We use the original ordered pairs for training and all 4,612 fine-grained pairs for testing (Sec. 3.1). We outperform all methods for three of the four attributes. For the two more objective attributes, *open* and *pointy*, our gains are sizeable—14% over Global for *open*. We attribute this to their localized nature, which is accurately captured by our learned metrics. No matter how fine-grained the difference is, it usually comes down to the top of the shoe (*open*) or the tip of the shoe (*pointy*). On the other hand, the subjective attributes are much less localized. The most challenging one is *comfort*, where our method performs slightly worse than Global, in spite of being better on the coarser pairs (Table 4.1a). We think this is because the locations of the subtleties vary greatly per pair.

Table 4.2: Accuracy comparison for the OSR scenes dataset. FG-LocalPair denotes the proposed approach.

	Natural	Open	Perspective	LgSize	Diagonal	ClsDepth
RelTree [69]	95.24	92.39	87.58	88.34	89.34	89.54
Global [84]	95.03	90.77	86.73	86.23	86.50	87.53
RandPair	92.97	89.40	84.80	84.67	84.27	85.47
LocalPair	94.63	93.27	88.33	89.40	90.70	89.53
FG-LocalPair	95.70	94.10	90.43	91.10	92.43	90.47

Table 4.3: Accuracy comparison for the PubFig faces dataset.

	Male	White	Young	Smiling	Chubby	Head	Brow	Eye	Nose	Lip	Face
RelTree [69]	85.33	82.59	84.41	83.36	78.97	88.83	81.84	83.15	80.43	81.87	86.31
Global [84]	81.80	76.97	83.20	79.90	76.27	87.60	79.87	81.67	77.40	79.17	82.33
RandPair	74.43	65.17	74.93	73.57	69.00	84.00	70.90	73.70	66.13	71.77	73.50
LocalPair	81.53	77.13	83.53	82.60	78.70	89.40	80.63	82.40	78.17	79.77	82.13
FG-LocalPair	91.77	87.43	91.87	87.00	87.37	94.00	89.83	91.40	89.07	90.43	86.70

4.3.3 Scenes and PubFig Results

We now shift our attention to OSR and PubFig, two commonly used datasets for relative attributes [59, 69, 84]. As described in Chapter 3, the paired supervision for these datasets originates from category-wise comparisons [84], and as such there are many more training pairs—on average over 20,000 per attribute.

Tables 4.2 and 4.3 show the accuracy for OSR and PubFig, respectively. Figure 4.6 and 4.7 show the individual precision recall curves for all attributes, using $|R_{\mathcal{A}}(\mathbf{x}_i) - R_{\mathcal{A}}(\mathbf{x}_j)|$ as a measure of confidence. On both datasets, our method outperforms all the baselines. Most notably, it outperforms RelTree [69], which at the time of our contribution was the very best accuracy reported on these datasets. This particular result is compelling not only because we improve the state-of-the-art at the time, but also because RelTree is a non-linear ranking function. Hence, we see that local learning with linear models is performing better than global learning with a non-linear model. With a lower capacity model, but the “right” training examples, the comparison is better learned. Our advantage over the global Relative Attributes linear model [84] is even greater.

On OSR, RandPair comes close to Global. One possible cause is the weak supervision from the category-wise constraints. While there are 20,000 pairs, they are less diverse. Therefore, a random sampling of 100 neighbors seems to reasonably mimic the performance when using all pairs. In contrast, our method is consistently stronger,

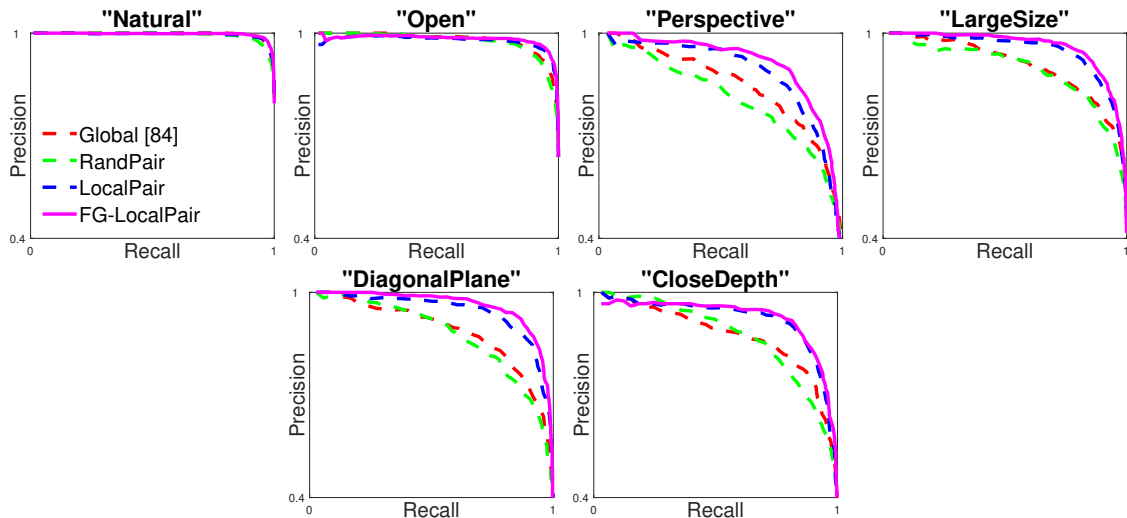


Figure 4.6: Precision-recall curves for OSR.

showing the value of our learned neighborhood pairs.

While metric learning (ML) is valuable across the board (FG-LocalPair > LocalPair), it has more impact on PubFig than OSR. We attribute this to PubFig’s more localized attributes. Subtle differences are what makes fine-grained comparison tasks hard. ML discovers the features behind those subtleties *with respect to each attribute*. Those features could be spatially localized regions or particular image cues (GIST vs. color). Indeed, our biggest gains compared to LocalPair (9% or more) are on *white*, where we learn to emphasize color bins, or *eye/nose*, where we learn to emphasize the GIST cells for the part regions. In contrast, the LocalPair method compares the face images as a whole, and is liable to find images of the same person as more relevant, regardless of their properties in that image (Fig. 4.3).

4.4 Predicting Useful Neighborhoods

As we have seen above, the goal of local learning is to tailor the model to the properties of the data surrounding the test instance. However, so far, like other prior work in local learning we have made an important core assumption: that the instances most *useful* for building a local model are those that are *nearest* to the test example. This assumption is well-motivated by the factors discussed above, in terms of data density and intra-class variation. Furthermore, as we saw above, identifying training examples solely based on proximity has the appeal of permitting specialized similarity functions (whether learned

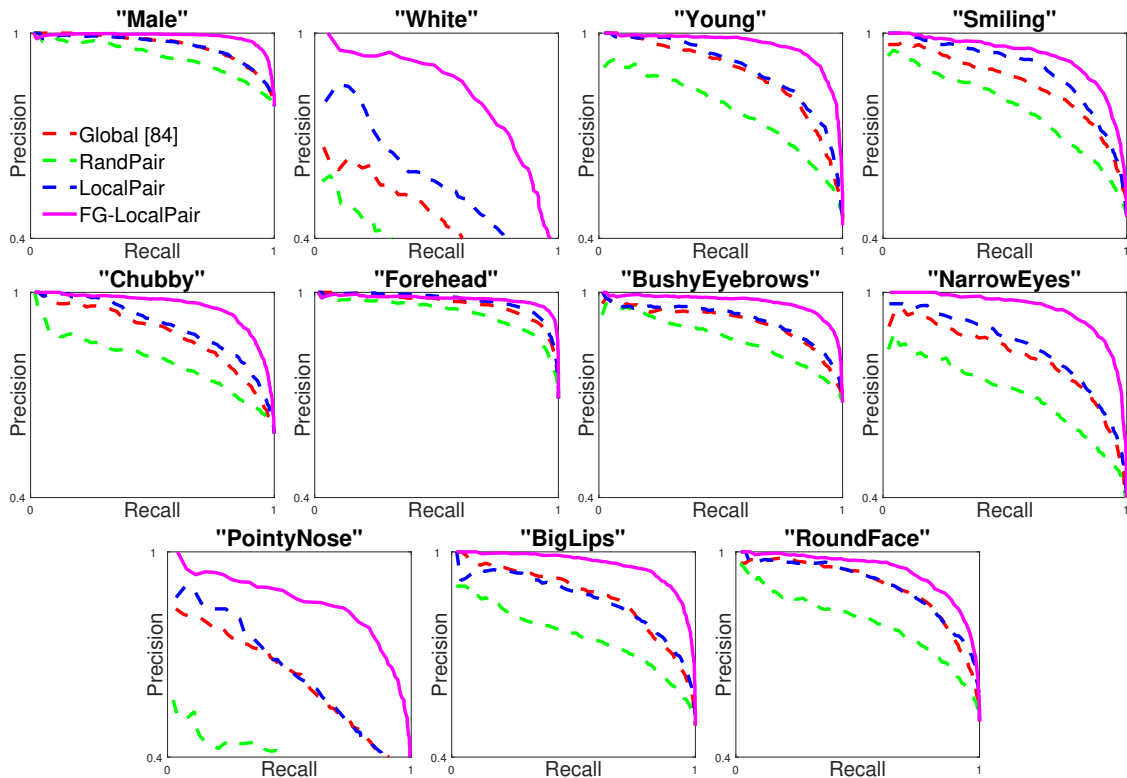


Figure 4.7: Precision-recall curves for PubFig.

or engineered for the problem domain), which can be valuable for good results, especially in structured input spaces.

On the other hand, there is a problem with this core assumption. By treating the individual nearness of training points as a metric of their utility for local training, existing methods fail to model how those training points will actually be employed. Namely, the relative success of a locally trained model is a function of the entire *set* or *distribution* of the selected data points—not simply the individual pointwise nearness of each one against the query. In other words, the ideal target subset consists of a set of instances that together yield a good predictive model for the test instance. Thus, local neighborhood selection ought to be considered jointly among training points.

Based on this observation, we explore ways to *learn* the properties of a “good neighborhood”.² We cast the problem in terms of large-scale multi-label classification, where we learn a mapping from an individual instance to an indicator vector over the

²This section expands on the neighbor selection approach described in Section 4.2.2, briefly summarizing our NIPS 2014 paper [126]. Please see that paper for full details.

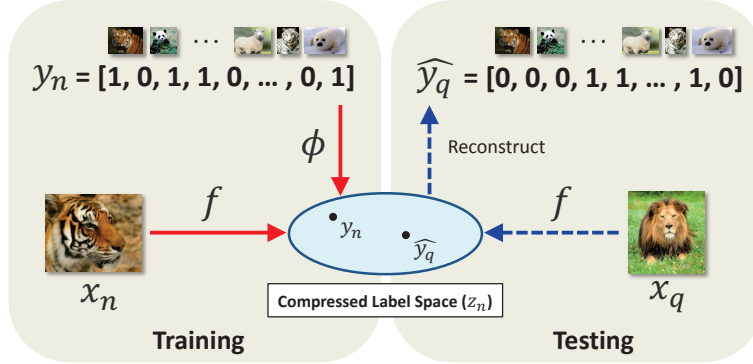


Figure 4.8: Overview of our compressed sensing based approach for predicting the most optimal *neighborhood* for learning a local classification model. \mathbf{y}_n and $\hat{\mathbf{y}}_q$ represent the M -dimensional neighborhood indicator vectors for a training and testing instance, respectively. ϕ is a $D \times M$ random matrix where D denotes the compressed indicators’ dimensionality. f is the learned regression function used to map the original image feature space to the compressed label space. By reconstructing back to the full label space, we get an estimate of $\hat{\mathbf{y}}_q$ indicating which labeled training instances together will form a good neighborhood for the test instance \mathbf{x}_q . This idea was published in NIPS 2014 [126].

entire training set that specifies which instances are jointly useful to the query. The approach maintains an inherent bias towards neighborhoods that are local, yet makes it possible to discover subsets that (i) deviate from a strict nearest-neighbor ranking and (ii) vary in size. We stress that learning what a good *neighbor* looks like (metric learning’s goal) is distinct from learning what a good *neighborhood* looks like (our goal). Whereas a metric can be trained with pairwise constraints indicating what should be near or far, jointly predicting the instances that ought to compose a neighborhood requires a distinct form of learning. The overall pipeline includes three main phases, shown in Figure 4.8.

4.4.1 Generating Neighborhoods

First, we devise an empirical approach to generate ground truth training neighborhoods $(\mathbf{x}_n, \mathbf{y}_n)$ that consist of an individual instance \mathbf{x}_n paired with a set of training instance indices capturing its target “neighbors”, the latter being represented as a M -dimensional indicator vector \mathbf{y}_n , where M is the number of labeled training instances. If $\mathbf{y}_n(j) = 1$, this means \mathbf{x}_j appears in the target neighborhood for \mathbf{x}_n . Otherwise, $\mathbf{y}_n(j) = 0$. We will generate N such pairs, where typically $N \ll M$.

As discussed in this chapter, there are good motivations for incorporating nearby points for local learning. Indeed, we do not intend to eschew the “locality” aspect of local learning. Rather, we start from the premise that points near to a query are

likely relevant—but relevance is not necessarily preserved purely by their rank order, nor must the best local set be within a fixed radius of the query (or have a fixed set size). Instead, we aim to generalize the locality concept to *jointly* estimate the members of a neighborhood such that *taken together* they are equipped to train an accurate query-specific model.

4.4.2 Model Learning using Compressed Sensing

Next, we pose the learning task as a large-scale multi-label classification problem. In multi-label classification, a single data point may have multiple labels. In our case, rather than predict which labels to associate with a novel example, we want to predict which training instances belong in its neighborhood. This is exactly what is encoded by the target indicator vectors defined above, \mathbf{y}_n . Furthermore, we want to exploit the fact that, compared to the number of all labeled training images, the most useful local neighborhoods will contain relatively few examples.

For this task, we adopt the Bayesian compressed sensing approach of [52] into our framework. With it, we can leverage sparsity in the high-dimensional target neighborhood space to efficiently learn a prediction function that jointly estimates all useful neighbors. To begin, for each of the N training neighborhoods, we project its M -dimensional neighborhood vector \mathbf{y}_n to a lower-dimensional space using a random transformation: $\mathbf{z}_n = \phi \mathbf{y}_n$, where ϕ is a $D \times M$ random matrix, and D denotes the compressed indicators’ dimensionality. Then, we learn regression functions to map the original features to these projected values $\mathbf{z}_{n_1}, \dots, \mathbf{z}_{n_N}$ as targets. That is, we obtain a series of $D \ll M$ regression functions f_1, \dots, f_D minimizing the loss in the compressed indicator vector space.

4.4.3 Inferring Neighborhoods

Finally, given a test instance \mathbf{x}_q , those same regression functions are applied to map to the reduced space, $[f_1(\mathbf{x}_q), \dots, f_D(\mathbf{x}_q)]$. We predict its complete neighborhood indicator vector $\hat{\mathbf{y}}_q$ by recovering the M -dimensional vector using a standard reconstruction algorithm from the compressed sensing literature. We use this neighborhood of points to train a classifier on the fly, which in turn is used to categorize \mathbf{x}_q .³

³Note that the neighborhood learning idea has been tested thus far only for classification tasks, though in principle applies similarly to ranking tasks.

In [126] we show substantial advantages over existing local learning strategies, particularly when attributes are multi-modal and/or its similar instances are difficult to match based on global feature distances alone. Our results illustrate the value in estimating the size and composition of discriminative neighborhoods, rather than relying on proximity alone. See our NIPS paper for the full details [126].

4.5 Discussion

In this chapter, I proposed a fine-grained local learning-to-rank approach based on analogous training comparisons and attribute-specific metric learning. The proposed approach demonstrated improvements over the state-of-the-art (at the time of publication) on three relative attributes datasets from three distinct domains: shoes, faces, and scenes. Even though the local rankers used only a small subset of the training pairs for learning, they were able to outperform the existing global models, thus supporting our hypothesis that learning from the “right” data can be more important than learning from “more” data.

However, in spite of my initial success with local rankers, there are some overarching challenges thus far unaddressed in this work. Firstly, like all prior work, I have ignored all test cases where the images are indistinguishable from each other. Equality prediction is a non-trivial task, as we shall see in Chapter 5, and is especially important for fine-grained tasks where the differences are subtle. Secondly, the pairwise nature of the supervision labels means that the space of all possible comparisons scales quadratically with the total number of images, which soon becomes intractable. Even after spending over \$1000, we have only collected less than 0.1% of all possible comparisons from our UT-Zap50K dataset. My proposed dense supervision approach from Chapter 6 and 7 addresses this issue indirectly. Thirdly, learning local models on the fly, though more accurate for fine-grained attributes, does come at a higher computational cost during runtime. However, there are straightforward ways to improve the run-time. For example, we could cluster the training pairs, build a local model for each cluster, and invoke the suitable model based on a test pair’s similarity to the cluster representatives. Lastly, due to the model’s reliance on analogous neighboring pairs, predictions on sparsely populated regions would perform poorly due to the lack of quality neighbors.

Chapter 5

Just Noticeable Attribute Differences

Having established the strength of local learning for fine-grained attribute comparisons, we now turn to task of predicting when a comparison is even possible. While some pairs of images have a clear ordering for an attribute (recall Fig. 1.2), for others the difference may be indistinguishable to human observers. Attempting to map relative attribute ranks to equality predictions is non-trivial, particularly since the span of indistinguishable pairs in an attribute space may vary in different parts of the feature space. In fact, as discussed above, despite the occasional use of unordered pairs for training¹, it is assumed in prior work that all test images will be orderable. However, the real-valued output of a ranking function as trained in Section 4.1.1 will virtually never be equal for two distinct inputs. Therefore, even though existing methods may learn to produce similar rank scores, it is unclear how to determine when a novel pair is “close enough” to be considered un-orderable.

We argue that this situation calls for a model of *just noticeable difference* among attributes. Just noticeable difference (JND) is a concept from psychophysics. It refers to the amount a stimulus has to be changed in order for it to be detectable by human observers at least half the time. For example, JND is of interest in color perception (which light sources are perceived as the same color?) and image quality assessment (up to what level of compression do the images look ok?). JNDs are determined empirically through tests of human perception. For example, JND in color can be determined by gradually altering the light source just until the human subject detects that the color has changed [31]. For this work, we borrow the JND term here as a loose analogy to the fine-grained problem we have on-hand.

Why is it challenging to develop a computational model of JND for relative attributes? At a glance, one might think it amounts to learning an optimal threshold on the difference of predicted attribute strengths. However, this begs the question of

¹Empirically, we found the inclusion of unordered pairs during training in [84] to have negligible impact at test time.

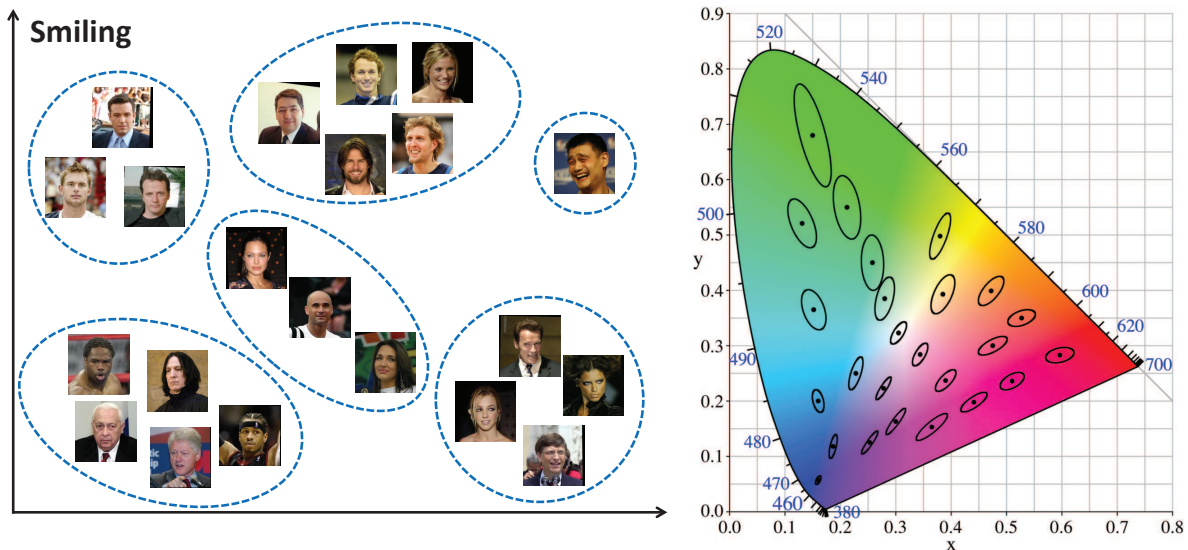


Figure 5.1: Analogous to the MacAdam ellipses in the CIE x,y color space (right) [31], relative attribute space is likely not uniform (left). That is, the regions within which attribute differences are indistinguishable may vary in size and orientation across the high-dimensional visual feature space. Here we see the faces within each “equally *smiling*” cluster exhibit varying qualities for differentiating smiles—such as age, gender, and visibility of the teeth—but are still difficult or impossible to order in terms of *smiling-ness*. As a result, simple metrics and thresholds on attribute differences are insufficient to detect subtle differences, as we will see in Section 5.2.2.

how one might properly and densely sample real images of a complex attribute (like *seriousness*) to gradually walk along the spectrum, so as to discover the right threshold with human input. More importantly, an attribute space need not be *uniform*. That is, depending on where we look in the feature space, the magnitude of attribute difference required to register a perceptible change may vary. Therefore, the simplistic “global threshold” idea falls short. Analogous issues also arise in color spaces, e.g., the famous MacAdam ellipses spanning indistinguishable colors in the CIE x,y color space vary markedly in their size and orientation depending on where in the feature space one looks (leading to the crafting of color spaces like CIE Lab that are more uniform). See Figure 5.1.

In this chapter, we introduce a solution to infer when two images are indistinguishable for a given attribute. Continuing with the theme of local learning, we develop a Bayesian approach that relies on *local* statistics of orderability. First, we construct a predicted attribute space using the standard RankSVM framework (Sec. 4.1.1). Then, on top of that model, we combine a likelihood computed in the predicted attribute space (Sec. 5.1.2) with a local prior computed in the original image feature space (Sec. 5.1.3).

We show our approach’s superior performance compared to various baselines for detecting noticeable differences, as well as demonstrate how attribute JND has potential benefits for an image search application (Sec. 5.2).

The work in this chapter was published in ICCV 2015 [127] and a book chapter [124].

5.1 Approach

The most straightforward approach to infer whether a novel image pair is distinguishable would be to impose a threshold on their rank differences, i.e., to predict “indistinguishable” if $|R_{\mathcal{A}}(\mathbf{x}_r) - R_{\mathcal{A}}(\mathbf{x}_s)| \leq \epsilon$. The problem is that unless the rank space is uniform, a global threshold ϵ is inadequate. In other words, the rank margin for indistinguishable pairs need not be constant across the entire feature space. By testing multiple variants of this basic idea, our empirical results confirm this is indeed an issue, as we will see in Section 5.2.

Our key insight is to formulate distinguishability prediction in a probabilistic, local learning manner. Mindful of the non-uniformity of relative attribute space, our approach uses distributions tailored to the data in the proximity of a novel test pair. Furthermore, we treat the relative attribute ranks as an imperfect mid-level representation on top of which we can learn to target the actual (sparse) human judgments about distinguishability. Our approach leverages both a low-level visual descriptor space, within which image pair proximity is learned, as well as a mid-level visual attribute space, within which attribute distinguishability is represented (Fig. 5.2). Whereas past ranking models have attempted to integrate equality into *training*, none attempt to distinguish between orderable and un-orderable pairs at test time.

5.1.1 Local Bayesian Model of Distinguishability

Let $D \in \{0, 1\}$ be a binary random variable representing the distinguishability of an image pair. For a distinguishable pair, $D = 1$. Given a novel test pair $(\mathbf{x}_r, \mathbf{x}_s)$, we are interested in the posterior:

$$P(D|\mathbf{x}_r, \mathbf{x}_s) \propto P(\mathbf{x}_r, \mathbf{x}_s|D)P(D), \quad (5.1)$$

to estimate how likely two images are distinguishable. To make a hard decision we take

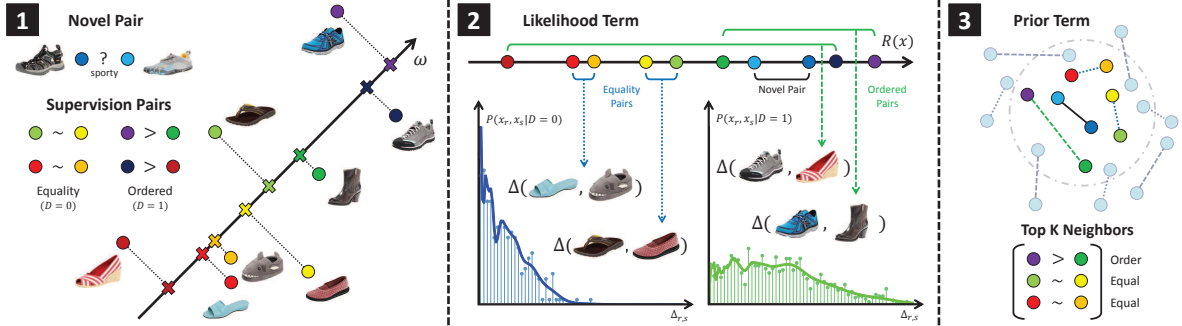


Figure 5.2: Overview of our Bayesian approach. (1) Learn a ranking function $R_{\mathcal{A}}$ using all annotated training pairs. (2) Estimate the likelihood densities of the equal and ordered pairs, respectively, using the pairwise distances in relative attribute space. (3) Determine the local prior by counting the labels of the analogous pairs in the image descriptor space. (4) Combine the results to predict whether the novel pair is distinguishable (not depicted). Best viewed on PDF.

the maximum a posteriori estimate over the two classes:

$$d^* = \arg \max_d P(D = d | \mathbf{x}_r, \mathbf{x}_s). \quad (5.2)$$

At test time, our method performs a two-stage cascade. If the test pair appears distinguishable, we return the response “more” or “less” according to whether $R_{\mathcal{A}}(\mathbf{x}_r) < R_{\mathcal{A}}(\mathbf{x}_s)$ (where R is trained in either a global or local manner). Otherwise, we say the test pair is indistinguishable. In this way we unify relative attributes with JND, generating partially ordered predictions in spite of the ranker’s inherent totally ordered outputs.

Next, we derive models for the likelihood and prior in Equation 5.1, accounting for the challenges described above.

5.1.2 The Likelihood Model

We use a kernel density estimator (KDE) to represent the distinguishability likelihood over image pairs. The likelihood captures the link between the observed rank differences and the human-judged just noticeable differences.

Let $\Delta_{r,s}$ denote the difference in attribute ranks for images r and s :

$$\Delta_{r,s} = |R_{\mathcal{A}}(\mathbf{x}_r) - R_{\mathcal{A}}(\mathbf{x}_s)|. \quad (5.3)$$

Recall that \mathcal{P}_o and \mathcal{P}_e refer to the sets of ordered and equal training image pairs, respectively. We compute the rank differences for all training pairs in \mathcal{P}_o and \mathcal{P}_e , and fit a non-parametric Parzen density:

$$P(\mathbf{x}_r, \mathbf{x}_s|D) = \frac{1}{|\mathcal{P}|} \sum_{i,j \in \mathcal{P}} K_h(\Delta_{i,j} - \Delta_{r,s}), \quad (5.4)$$

for each set in turn. Here \mathcal{P} refers to the ordered pairs \mathcal{P}_o when representing distinguishability ($D = 1$), and the equal pairs \mathcal{P}_e when representing indistinguishability ($D = 0$). The Parzen density estimator [86] superimposes a kernel function K_h at each data pair. In our implementation, we use Gaussian kernels. It integrates local estimates of the distribution and resists overfitting. The KDE has a smoothing parameter h that controls the model complexity. To ensure that all density is contained within the positive absolute margins, we apply a positive support to the estimator. Namely, we transform $\Delta_{i,j}$ using a log function, estimate the density of the transformed values, and then transform back to the original scale. See (2) in Figure 5.2.

The likelihood reflects how well the equal and ordered pairs are separated in the attribute space. However, critically, $P(\mathbf{x}_r, \mathbf{x}_s|D = 1)$ need not decrease monotonically as a function of rank differences. In other words, the model permits returning a higher likelihood for certain pairs separated by smaller margins. This is a direct consequence of our choice of the non-parametric KDE, which preserves local models of the original training data. This is valuable for our problem setting because in principle it means our method can correct imperfections in the original learned ranks and account for the non-uniformity of the space.

5.1.3 The Prior Model

Finally, we need to represent the prior over distinguishability. The prior could simply count the training pairs, i.e., let $P(D = 1)$ be the fraction of all training pairs that were distinguishable. However, we again aim to account for the non-uniformity of the visual feature space. Thus, we estimate the prior based only on a subset of data near the input images. Intuitively, this achieves a simple prior for the label distribution in multiple pockets of the feature space:

$$P(D = 1) = \frac{1}{K} |\mathcal{P}'_o|, \quad (5.5)$$

where $\mathcal{P}'_o \subset \mathcal{P}_o$ denotes the set of K neighboring ordered training pairs. $P(D = 0)$ is defined similarly for the indistinguishable pairs \mathcal{P}_e . Note that while the likelihood is

computed over the pair’s rank difference, the locality of the prior is with respect to the image descriptor space. See (3) in Figure 5.2.

To localize the relevant pocket of the image space, we adopt the metric learning strategy detailed in Section 4.2.2. Using the learned metric, pairs analogous to the novel input $(\mathbf{x}_r, \mathbf{x}_s)$ are retrieved based on a product of their individual Mahalanobis distances, so as to find pairs whose members both align.

5.2 Evaluation

We present results on the core JND detection task (Sect. 5.2.2) on two challenging datasets and demonstrate its impact for an image search application (Sect. 5.2.3).

5.2.1 Experimental Setup

Datasets and Ground Truth Our task requires attribute datasets that (1) have instance-level supervision, meaning annotators were asked to judge attribute comparisons on individual pairs of images, not object categories as a whole, and (2) have pairs labeled as “equal” and “more/less”. To train and evaluate just noticeable differences, we must have annotations of utmost precision. Therefore, we take extra care in establishing the (in)distinguishable ground truth for both datasets. For the following experiments, we use the high quality labels **UT-Zap50K-EQ** and **LFW-10-EQ** described in Sections 3.1 and 3.3 for the shoe and face domains respectively. Recall that these datasets have not only ordered pairs \mathcal{P}_o but also equal pairs \mathcal{P}_e .

Baselines We are the first to address the attribute JND task. No prior methods infer indistinguishability at test time [58, 69, 84, 95, 96]. Therefore, we develop multiple baselines to compare to our approach:

- **Rank Margin:** Use the magnitude of $\Delta_{r,s}$ as a confidence measure that the pair r, s is distinguishable. This baseline assumes the learned rank function produces a uniform feature space, such that a *global threshold* on rank margins would be sufficient to identify indistinguishable pairs. To compute a hard decision for this method (for F1-scores), we threshold the Parzen window likelihood estimated from the training pairs by ϵ , the mid-point of the likelihood means.

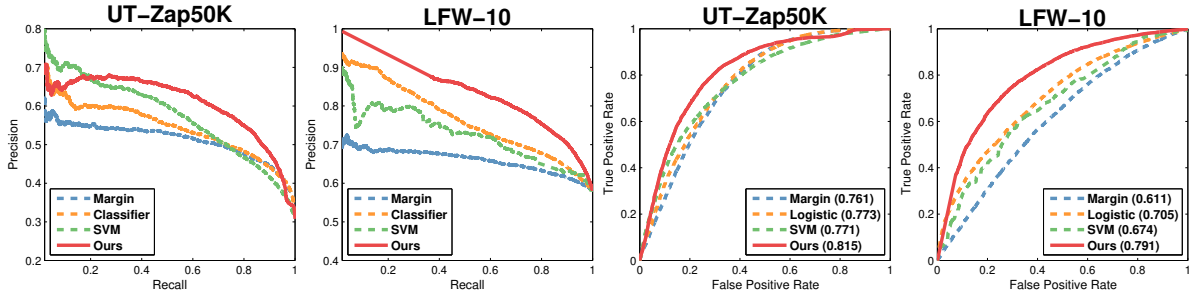


Figure 5.3: Just noticeable difference detection accuracy for all attributes. We show the precision-recall (left two) and ROC curves (right two) for the shoes (UT-Zap50k) and faces (LFW-10) datasets. Legends show AUC values for ROC curves. Note that the Mean Shift baseline does not appear here, since it does not produce confidence values.

- Logistic Classifier** [58]: Train a logistic regression classifier to distinguish training pairs in \mathcal{P}_o from those in \mathcal{P}_e , where the pairs are represented by their rank differences $\Delta_{i,j}$. To compute a hard decision, we threshold the posterior at 0.5. This is the method used in [58] to obtain a probabilistic measure of attribute equality. It is the closest attempt we can find in the literature to represent equality predictions, though the authors do not evaluate its accuracy. This baseline also maintains a global view of attribute space.
- SVM Classifier**: Train a nonlinear SVM classifier with a RBF kernel to distinguish ordered and equal pairs. We encode pairs of images as single points by concatenating their image descriptors. To ensure symmetry, we include training instances with the two images in either order.²
- Mean Shift**: Perform mean shift clustering on the predicted attribute scores $R_{\mathcal{A}}(\mathbf{x}_i)$ for all training images. Images falling in the same cluster are deemed indistinguishable. Since mean shift clusters can vary in size, this baseline does *not* assume a uniform space. Unlike our method, it fails to leverage distinguishability supervision as it processes the ranker outputs.

Implementation Details For UT-Zap50K, just like in Section 4.3.1, we use 960-dim GIST and 30-bin Lab color histograms as image descriptors. For LFW-10, they are 8,300-dim part-based features learned on top of dense SIFT bag of words features (provided by

²We also implemented other encoding variants, such as taking the difference of the image descriptors or using the predicted attribute scores $R_{\mathcal{A}}(x_i)$ as features, and they performed similarly or worse.

Table 5.1: JND detection on UT-Zap50K (F1 scores).

	Open	Pointy	Sporty	Comfort	All Attributes
Margin	48.95	67.48	66.93	57.09	60.11 \pm 1.89
Logistic	10.49	62.95	63.04	45.76	45.56 \pm 4.13
SVM	48.82	50.97	47.60	40.12	46.88 \pm 5.73
Mean Shift	54.14	58.23	60.76	61.60	58.68 \pm 8.01
Ours	62.02	69.45	68.89	54.63	63.75 \pm 3.02

Table 5.2: JND detection on LFW-10 (F1 scores). NaN occurs when recall=0 and precision=inf.

	Bald	Hair	Eyes	Look	Masc.	Mouth	Smile	Teeth	Head	Young	All Attributes
Margin	71.10	55.81	74.16	61.36	82.38	62.89	60.56	65.26	67.49	34.20	63.52 \pm 2.67
Logistic	75.77	53.26	86.71	64.27	87.29	63.41	59.66	64.83	75.00	NaN	63.02 \pm 1.84
SVM	79.06	32.43	89.70	70.98	87.35	70.27	55.01	39.09	79.74	NaN	60.36 \pm 9.81
Mean Shift	66.37	56.69	54.50	51.29	69.73	68.38	61.34	65.73	73.99	23.19	59.12 \pm 10.51
Ours	81.75	69.03	89.59	75.79	89.86	72.69	73.30	74.80	80.49	32.89	74.02 \pm 1.66

the authors). We reduce their dimensionality to 100 with PCA to prevent overfitting. The part-based features [96] isolate localized regions of the face (e.g., exposing cues specific to the eyes vs. hair). We experimented with both linear and RBF kernels for R_A . Since initial results were similar, we use linear kernels for efficiency. We use Gaussian kernels for the Parzen windows. We set all hyperparameters (h for the KDE, bandwidth for Mean Shift, K for the prior) on held-out validation data. To maximize the use of training data, in all results below, we use leave-one-out evaluation and report results over 4 folds of random training-validation splits.

5.2.2 Just Noticeable Difference Detection

We evaluate just noticeable difference detection accuracy for all methods on both datasets. Figure 5.3 shows the precision-recall curves and ROC curves, where we pool the results from all 4 and 10 attributes in UT-Zap50K and LFW-10, respectively. Tables 5.1 and 5.2 report the summary F1-scores and standard deviations for each individual attribute. The F1-score is a useful summary statistic for our data due to the unbalanced nature of the test set: 25% of the shoe pairs and 80% of the face pairs are indistinguishable for some attribute.

Overall, our method outperforms all baselines. We obtain sizeable gains—roughly 4-18% on UT-Zap50K and 10-15% on LFW-10. This clearly demonstrates the advantages of our local learning approach, which accounts for the non-uniformity of attribute

	Indistinguishable				Distinguishable	
Pointy						
Sporty						
Big Eyes						
Smiling						
Error Cases						

Figure 5.4: Example predictions. The top four rows are pairs our method correctly classifies as indistinguishable (left panel) and distinguishable (right panel), whereas the Rank Margin baseline fails. Each row shows pairs for a particular attribute. The bottom row shows failure cases by our method; i.e., the bottom left pair is indistinguishable for pointiness, but we predict distinguishable.

space. The “global approaches”, Rank Margin and Logistic Classifier, reveal that a uniform mapping of the relative attribute predictions is insufficient. In spite of the fact that they include equal pairs during training, simply assigning similar scores to indistinguishable pairs is inadequate. Their weakness is likely due both to noise in those mid-level predictions as well as the existence of JND regions that vary in scale. Furthermore, the results show that even for challenging, realistic image data, we can identify just noticeable differences at a high precision and recall, up to nearly 90% in some cases.

The SVM baseline is much weaker than our approach, indicating that discriminatively learning what indistinguishable image pairs look like is insufficient. This result underscores the difficulty of learning subtle differences in a high-dimensional image descriptor space, and supports our use of the compact rank space for our likelihood model. Looking at the per-attribute results (Tables 5.1 and 5.2), we see that our method also outperforms the Mean Shift baseline. While Mean Shift captures dominant clusters in the spectrum of predicted attribute ranks for certain attributes, for others (like *pointy* or *masculine*) we find that the distribution of output predictions are more evenly spread. Despite the fact that the rankers are optimized to minimize margins for equal pairs, simple post-processing of their outputs is inadequate.

Figure 5.4 shows qualitative prediction examples. Here we see the subtleties of



Figure 5.5: Example just noticeable differences. In each row, we take leftmost image as a starting point, then walk through nearest neighbors in relative attribute space until we hit an image that is distinguishable, as predicted by our method. For example, in row 2, our method finds the left block of images to be indistinguishable for *sportiness*; it flags the transition from the flat dress shoe to the pink “loafer-like sneaker” as being a noticeable difference.

JND. Whereas past methods would be artificially forced to make a comparison for the left panel of image pairs, our method declares them indistinguishable. Pairs may look very different overall (e.g., different hair, race, headgear) yet still be indistinguishable *in the context of a specific attribute*. Meanwhile, those that are distinguishable (right panel) may have only subtle differences.

Figure 5.5 illustrates examples of just noticeable difference “trajectories” computed by our method. We see how our method can correctly predict that various instances are indistinguishable, even though the raw images can be quite diverse (e.g., a strappy sandal and a flat dress shoe are equally *sporty*). Similarly, it can detect a difference even when the image pair is fairly similar (e.g., a lace-up sneaker and smooth-front sneaker are distinguishable for *openness* even though the shapes are close).

Figures 5.6 and 5.7 display 2D t-SNE [72] embeddings for a subset of 5,000 shoe images based on the original image feature space and our learned attribute space for the attribute *pointy*. To compute the embeddings for our method, we represent each image \mathbf{x}_i by its posterior probabilities of being indistinguishable to every other image. i.e. $v(\mathbf{x}_i) = [P(D = 0|\mathbf{x}_i, \mathbf{x}_1), P(D = 0|\mathbf{x}_i, \mathbf{x}_2), \dots, P(D = 0|\mathbf{x}_i, \mathbf{x}_N)]$ where N is the total number of images in the embedding. We see that while the former produces a rather evenly distributed mapping without distinct structures, the latter produces a mapping containing unique structures along with “pockets” of indistinguishable images. Such structures precisely reflect the non-uniformity we anticipated in Figure 5.1.

5.2.3 Image Search Application

Finally, we demonstrate how JND detection can enhance an image search application. Specifically, we incorporate our model into the WhittleSearch framework of Kovashka

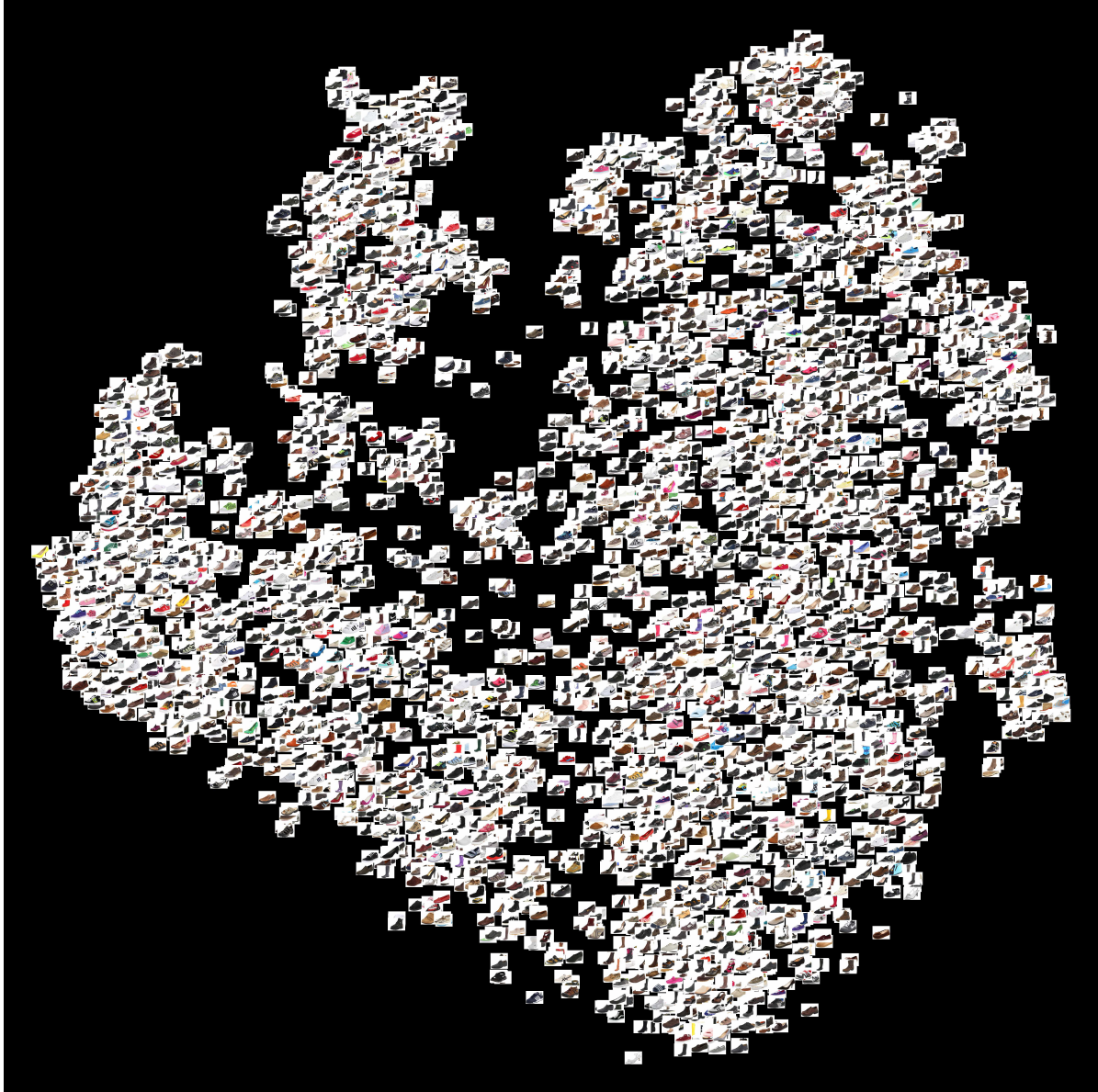


Figure 5.6: t-SNE visualization of the original feature space. See Figure 5.7 for the visualization of our learned attribute space. Best viewed on PDF.



Figure 5.7: t-SNE visualization of our learned attribute space for the attribute *pointy*. Shoes with a similar level of *pointiness* are placed closer together in our learned space, forming loose “pockets” of indistinguishability. Best viewed on PDF.



Figure 5.8: The modified WhittleSearch framework. The user can now express an “equality” feedback, speeding up the process of finding his envisioned target.

et al. [59]. WhittleSearch is an interactive method that allows a user to provide relative attribute feedback, e.g., by telling the system that he wants images “more *sporty*” than some reference image. The method works by intersecting the relative attribute constraints, scoring database images by how many constraints they satisfy, then displaying the top scoring images for the user to review. See [59] for details.

We augment that pipeline such that the user can express not only “more/less” preferences, but also “equal” preferences (Fig. 5.8). For example, the user can now say, “I want images that are equally *sporty* as image x .” Intuitively, enriching the feedback in this manner should help the user more quickly zero in on relevant images that match his envisioned target. To test this idea, we mimic the method and experimental setup of [59] as closely as possible, including their feedback generation simulator.

We evaluate a proof-of-concept experiment on UT-Zap50K, which is large enough to allow us to sequester disjoint data splits for training our method and performing the searches (LFW-10 is too small). We select 200 images at random to serve as the mental targets a user wants to find in the database, and reserve 5,000 images for the database. The user is shown 16 reference images and expresses 8 feedback constraints per iteration.

Figure 5.9 shows the results. Following [59], we measure the relevance rank of the target as a function of feedback iterations (left, lower is better), as well as the similarity of all top-ranked results compared to the target (right, higher is better). We see that JNDs substantially bolster the search task. In short, the user gets to the target in fewer iterations because he has a more complete way to express his preferences—and the system understands what “equally” means in terms of attribute perception.

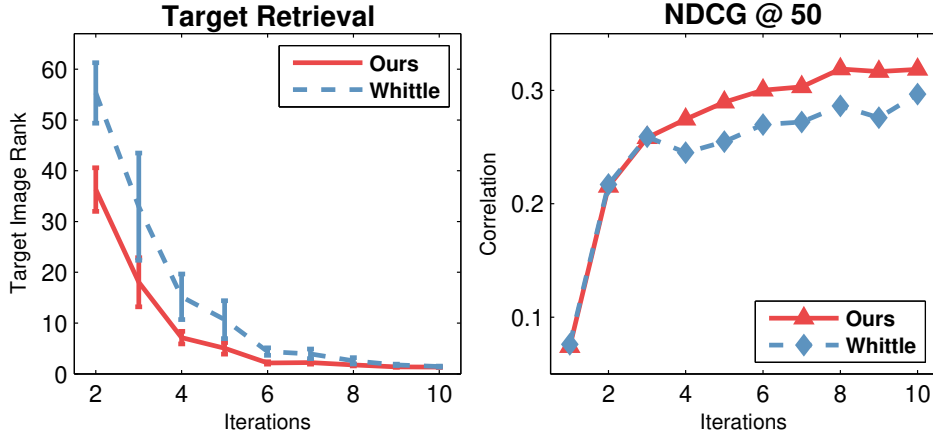


Figure 5.9: Image search results. We enhance an existing relative attribute search technique called WhittleSearch [59] with our JND detection model. The resulting system finds target images more quickly (left) and produces a better overall ranking of the database images (right).

5.3 Discussion

In this chapter, I explored the task of identifying perceivable differences between two highly similar images, with respect to a given attribute. Motivated by my local learning approach in Chapter 4, I proposed a Bayesian prediction model that leverages the local statistics of orderability to distinguish JNDs in attributes. Aside from being the first to address this issue of “equality” in a fine-grained comparison setting, I was also able to demonstrate the effectiveness of my proposed local probabilistic model over multiple alternative strategies in two distinct domains.

With that said, one potential weakness of our approach is its reliance on the large-margin ranker $R_{\mathcal{A}}$ as the intermediate representations of the individual images, which are used to compute the likelihood and the prior terms. As we saw in Chapter 4, the standard large-margin ranker is not perfect and a poorly trained initial $R_{\mathcal{A}}$ could harm all subsequent computations. Furthermore, just like our local learning approach, whenever the local model is used for $R_{\mathcal{A}}$, the computation cost at runtime could become unscalable as the number of training pairs increases.

Finally, one major issue that I encountered was the lack of relevant neighbors around sparsely supervised regions in the attribute space. Such a scenario would severely hinder the effectiveness of the approach since the closest neighbors are no longer representative of the actual local statistics. The underlying issue is that unlike in a true JND experiment from psychophysics, where for example, a light source can be systematically adjusted until the human detects a change, we do not have such fine control

over our training data. However, as we will see in Chapter 6 and 7, this is an issue that a generative model has the potential to address.

Chapter 6

Dense Supervision Through Semantic Jitter

Having discussed the algorithmic improvements over the last two chapters using local learning, we now look at an orthogonal issue of the *sparsity of supervision* for fine-grained comparisons. The sparsity of supervision problem consists of two areas: label availability and image availability.

Due to the pairwise nature of the supervision labels, the space of all possible comparisons is quadratic in the number of potential training images. This quickly makes it intractable to label an image collection exhaustively for its comparative properties. Furthermore, attribute comparisons also entail a greater cognitive load than, for example, object category labeling. Even for our UT-Zap50K dataset (Secs. 3.1 and 3.2), which is already the largest existing relative attribute dataset, only less than 0.1% of all possible comparisons have been labeled. The fact is that there is a major size gap between standard datasets labeled for classification (now in the millions [21]) and those for comparisons (at best in the thousands). A popular shortcut is to propagate category-level comparisons down to image instances [8, 84]—e.g., deem all ocean scenes as “more *open*” than all forest scenes—but this introduces substantial label noise and in practice underperforms training with instance-level comparisons [60].

Meanwhile, another parallel issue more insidious than the annotation cost is the problem of even curating training images that sufficiently illustrate fine-grained differences. Critically, sparse supervision arises not only because we lack resources to get enough image pairs labeled, but also because *we lack a direct way to curate photos demonstrating all sorts of subtle attribute changes*. In other words, the “right” data for fine-grained learning might not even be available in the existing training data. For example, how might we gather unlabeled image pairs depicting all subtle differences in “sportiness” in clothing images or “surprisedness” in faces? As a result, even today’s best datasets contain only partial representations of an attribute.

In this chapter, I propose an approach to “densify” supervision for fine-grained comparison tasks by leveraging synthetic images sampled from an attribute-conditioned

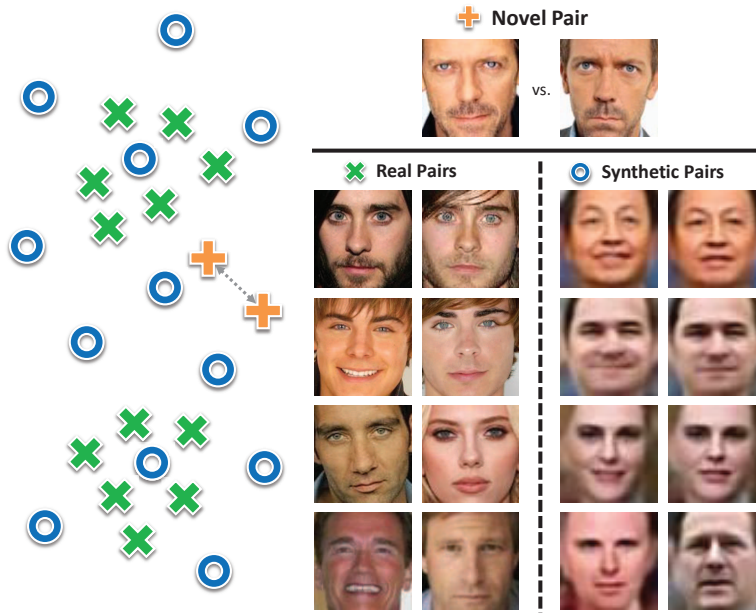


Figure 6.1: Our method “densifies” supervision for training ranking functions to make visual comparisons, by generating ordered pairs of synthetic images. Here, when learning the attribute *smiling*, real training images need not be representative of the entire attribute space (e.g., web photos may cluster around commonly photographed expressions, like toothy smiles). Our idea “fills in” the sparsely sampled regions to enable fine-grained supervision. Given a novel pair (top), the nearest synthetic pairs (right) may present better training data than the nearest real pairs (left).

generative model. These semantically jittered synthetic pairs can be used independently or in conjunction with existing image pairs. First, we describe the generative model (Sec. 6.1.1) and how we elicit dense supervision pairs from it (Sec. 6.1.2). Then, we integrate synthetic and real images to train rankers for attribute comparisons (Sec. 6.2). Lastly, to demonstrate our approach’s versatility, in experiments we explore both successful learning-to-rank models from the attributes literature previewed in Section 4.1 (Sec. 6.1.3).

The work in this chapter was published in ICCV 2017 [128].

6.1 Approach

The key to improving *coverage* in the attribute space is the ability to generate images exhibiting subtle differences—with respect to the given attribute—while keeping the others constant. In other words, we want to walk semantically in the high-level attribute space (Fig. 6.1).

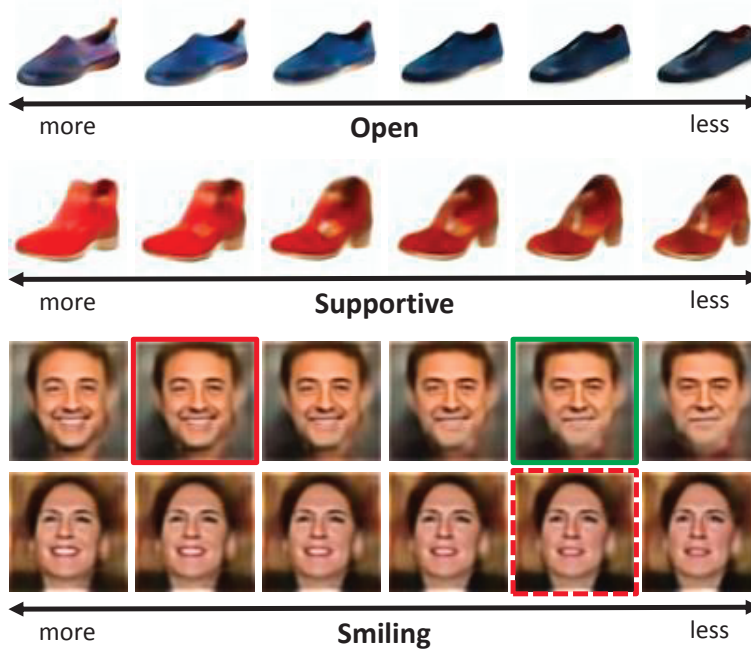


Figure 6.2: Spectra of generated images given an identity and an attribute. We form two types of image pairs: The two solid boxes represent an *intra-identity pair*, whereas the two red boxes represent an *inter-identity pair*.

6.1.1 Attribute-Conditioned Image Generator

We adopt an existing state-of-the-art image generation system, **Attr2Img** (Attr2Img), recently introduced by Yan et al. [118, 119], which can generate images that exhibit a given set of attributes and latent factors.

Suppose we have a lexicon of N_a attributes, $\{\mathcal{A}_1, \dots, \mathcal{A}_{N_a}\}$. Let $\mathbf{y} \in \mathbb{R}^{N_a}$ be a vector containing the strength of each attribute, and let $\mathbf{z} \in \mathbb{R}^{N_z}$ be the latent variables. The Attr2Img approach constructs a generative model for $p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z})$ that produces realistic images $\mathbf{x} \in \mathbb{R}^{N_x}$ conditioned on \mathbf{y} and \mathbf{z} . The authors maximize the variational lower bound of the log-likelihood $\log p_\theta(\mathbf{x}|\mathbf{y})$ in order to obtain the model parameters θ . The model is implemented with a Conditional Variational Auto-Encoder (CVAE). The network architecture generates the entangled hidden representation of the attributes and latent factors with multilayer perceptrons, then generates the image pixels with a coarse-to-fine convolutional decoder. The authors apply their approach for attribute progression, image completion, and image retrieval. See [118, 119] for more details.¹

¹We use the original model rather than the variant disCVAE [118] since the latter requires additional supervision in the form of foreground object masks.

6.1.2 Generating Dense Synthetic Image Pairs

We propose to leverage the Attr2Img [118] engine to supply realistic synthetic training images that “fill in” under-represented regions of image space, which we show helps train a model to infer attribute comparisons.

The next key step is to generate a series of synthetic *identities*, then sample images for those identities that are close by in a desired semantic attribute space.² The resulting images will comprise a set of synthetic image pairs \mathcal{S}_A . We explore two cases for using the generated pairs: one where their putative ordering is verified by human annotators, and another where the ordering implied by the generation engine is taken as their (noisy) label. Section 6.1.3 describes how we use the hybrid real and synthetic image pairs to train specific attribute predictors.

Each identity is defined by an entangled set of latent factors and attributes. Let $p(\mathbf{y})$ denote a prior over the attribute occurrences in the domain of interest. We model this prior with a multivariate Gaussian whose mean and covariance are learned from the attribute strengths observed in real training images: $p(\mathbf{y}) = \mathcal{N}(\mu, \Sigma)$. This distribution captures the joint interactions between attributes, such that a sample from the prior reflects the co-occurrence behavior of different pairs of attributes (e.g., shoes that are very pointy are often also uncomfortable, faces that have facial hair are often masculine, etc.).³ The prior over latent factors $p(\mathbf{z})$, captures all non-attribute properties like pose, background, and illumination. Following [119], we represent $p(\mathbf{z})$ with an isotropic multivariate Gaussian.

To sample an identity

$$\mathcal{I}_j = (\mathbf{y}_j, \mathbf{z}_j) \tag{6.1}$$

we sample \mathbf{y}_j and \mathbf{z}_j from their respective priors. Then, using an Attr2Img model trained for the domain of interest, we sample from $p_\theta(\mathbf{x}|\mathbf{y}_j, \mathbf{z}_j)$ to generate an image $\hat{\mathbf{x}}_j \in \mathbb{R}^{N_x}$ for this identity. Alternatively, we could sample an identity from a single real image,

²Note that here the word “identity” means an instance for some domain, not necessarily a human identity; in experiments we apply our idea both for human faces as well as fashion images of shoes.

³Note that this prior is nonetheless assumed to be coarse, since a subset of dimensions in \mathbf{y} consist of the very attributes we wish to learn better via densifying supervision. For the sake of the prior, the training image attribute strengths originate from the raw decision outputs of a preliminary binary attribute classifier trained on disjoint data labeled for the presence/absence of the attribute (see Sec. 6.2.1).

after inferring its latent variables through the generative model [122]. However, doing so requires having access to attribute labels for that image.

Next we modify the strength of a single attribute in \mathbf{y} while keeping all other variables constant. This yields two “tweaked” identities $\mathcal{I}_j^{(-)}$ and $\mathcal{I}_j^{(+)}$ that look much like \mathcal{I}_j , only with a bit less or more of the attribute, respectively. Specifically, let $\sigma_{\mathcal{A}}$ denote the standard deviation of attribute scores observed in real training images for attribute \mathcal{A} . We revise the attribute vector for identity \mathcal{I}_j by replacing the dimension for attribute \mathcal{A} according to

$$\begin{aligned} \mathbf{y}_j^{(-)}(\mathcal{A}) &= \mathbf{y}_j(\mathcal{A}) - 2\sigma_{\mathcal{A}} \text{ and} \\ \mathbf{y}_j^{(+)}(\mathcal{A}) &= \mathbf{y}_j(\mathcal{A}) + 2\sigma_{\mathcal{A}}, \end{aligned} \tag{6.2}$$

and $\mathbf{y}_j^{(-)}(a) = \mathbf{y}_j^{(+)}(a) = \mathbf{y}_j(a), \forall a \neq \mathcal{A}$. Finally, we sample from $p_{\theta}(\mathbf{x}|\mathbf{y}_j^{(-)}, \mathbf{z}_j)$ and $p_{\theta}(\mathbf{x}|\mathbf{y}_j^{(+)}, \mathbf{z}_j)$ to obtain images $\hat{\mathbf{x}}_j^{(-)}$ and $\hat{\mathbf{x}}_j^{(+)}$. Recall that our identity sampling accounts for inter-attribute co-occurrences. Slightly altering a *single* attribute is in line with our goal to densify supervision and recovers plausible but yet-unseen instances.

Figure 6.2 shows examples of synthetic images generated for a sampled identity, varying only in one attribute. The generated images form a smooth progression in the attribute space. This is exactly what allows us to curate fine-grained pairs of images that are very similar in attribute strength. Crucially, such pairs are rarely possible to curate systematically among real images. The exception is special “hands-on” scenarios, e.g., for faces, asking subjects in a lab to slowly exhibit different facial expressions, or systematically varying lighting or head pose (cf. PIE, Yale face datasets). The hands-on protocol is not only expensive, it is inapplicable in most domains outside of faces and for rich attribute vocabularies. Furthermore, the generation process allows us to collect in a controlled manner subtle visual changes *across* identities as well.

Next we pair up the synthetic images to form the set $\mathcal{S}_{\mathcal{A}}$, which, once (optionally) verified and pruned by human annotators, will augment the real training image pairs $\mathcal{P}_{\mathcal{A}}$. In order to maximize our coverage of the attribute space, we sample two types of synthetic image pairs: *intra-identity pairs*, which are images sampled from the same identity’s spectrum and *inter-identity pairs*, which are images sampled from different spectrums (Fig. 6.2).

We expect many of the generated pairs to be valid, meaning that both images are realistic and that the pair exhibits a slight difference in the attribute of interest.

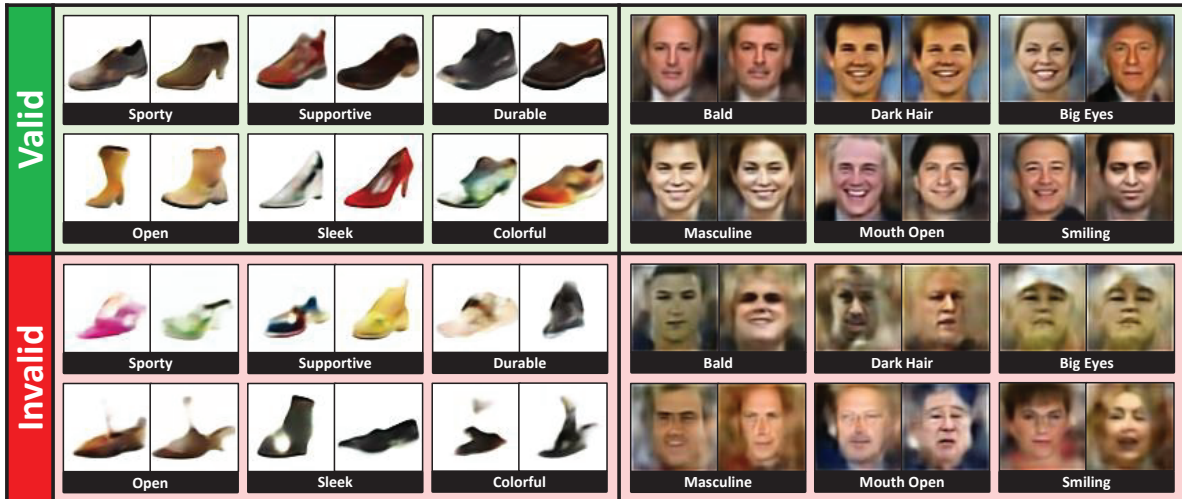


Figure 6.3: Synthetic image pairs generated by our semantic jitter approach (top), including some failure cases where the generator failed to synthesize realistic looking images (bottom). Here “valid” vs. “invalid” is as judged by five human annotators.

However, this need not always be true. In some cases the generator will create images that do not appear to manipulate the attribute of interest, or where the pair is close enough in the attribute to be indistinguishable, or where the images simply do not look realistic enough to tell (Fig. 6.3). Our experiments indicate this happens about 15% of the time.

To correct erroneous pairs, we collect order labels from 5 crowdworkers per pair. However, while human-verified pairs are most trustworthy for a learning algorithm, we suspect that even noisy (unverified) pairs could be beneficial too, provided the learning algorithm (1) has high enough capacity to accept a lot of them and/or (2) is label-noise resistant. Unverified pairs are attractive because they are free to generate in mass quantities. We examine both cases below.

6.1.3 Learning to Rank with Hybrid Comparisons

In principle any learning algorithm for visual comparisons could exploit the newly generated synthetic image pairs. We consider two common ones from the attribute literature: RankSVMs with local learning (from Chap. 4) and a deep Siamese RankNet with a spatial transformer network (STN), which build upon the basic deep learning formulation overviewed in Section 4.1.2.

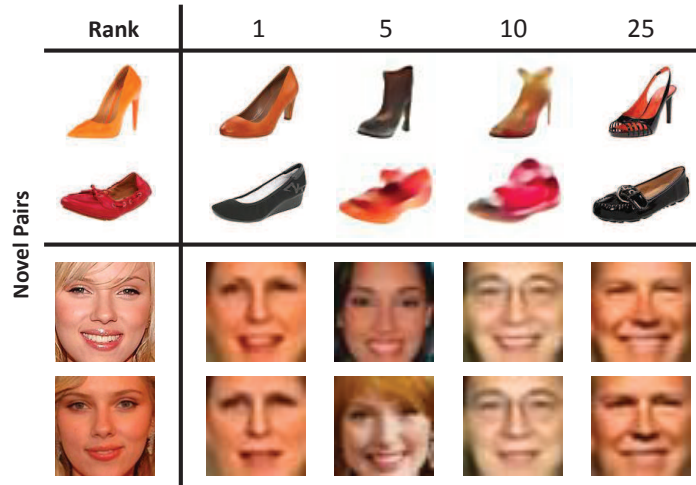


Figure 6.4: Examples of nearest neighbor image pairs given novel test pairs (left). Both real and synthetic image pairs appear in the top neighbors, suggesting their combined importance in the local learning algorithm.

RankSVM+Local Learning We employ RankSVM with a local learning model, exactly as we did in Chapter 4. Given a hybrid set of sparse real pairs and dense synthetic pairs, $\{\mathcal{P}_A \cup \mathcal{S}_A\}$, we use a local model to select the most relevant *mix* of real and synthetic pairs (Fig. 6.4). Just as bare bones nearest neighbors relies on adequate density of labeled exemplars to succeed, in general local learning is expected to flourish when the space of training examples is more densely populated. Thus, local learning is congruent with our hypothesis that data *density* is at least as important as data *quantity* for learning subtle differences (Fig. 6.1).

DeepCNN+Spatial Transformer Our choice for the second ranker is motivated both by its leading empirical performance [107] as well as its high capacity, which makes it data hungry. This deep learning to rank method combines a CNN optimized for a paired ranking loss [12] (briefly described in Sec. 4.1.2) together with a spatial transformer network (STN) [47]. In particular,

$$R_{\mathcal{A}}^{(cnn)}(\phi(\mathbf{x})) = \text{RankNet}_{\mathcal{A}}(\text{STN}(\phi(\mathbf{x}))), \quad (6.3)$$

where RankNet denotes a Siamese network with duplicate stacks. During training these stacks process ordered pairs, learning filters that map the images to scalars that preserve the desired orderings in \mathcal{P}_A . The STN is trained simultaneously to discover the localized patch per image that is most useful for ranking the given attribute (e.g., it may focus on

the mouth for *smiling*). Given a single novel image, either stack can be used to assign a ranking score. See [107] for details. As above, our approach trains this CNN with all pairs in $\{\mathcal{P}_A \cup \mathcal{S}_A\}$.

Generator vs. Ranker A natural question to ask is why not feed back the synthetic image pairs into the same generative model that produced them, to try and enhance its training? We avoid doing so for two important reasons. First, this would lead to a circularity bias where the system would essentially be trying to exploit new data that it has already learned to capture well (and hence could generate already). Second, the particular image generator we employ is not equipped to learn from relative supervision nor make relative *comparisons* on novel data. Rather, it learns from individual images with absolute attribute strengths. Thus, we use the synthetic data to train a distinct model capable of learning relative visual concepts.

Curating Images vs. Supervision While traditional data collection methods lack a direct way to curate image pairs covering the full space of attribute variations, our approach addresses exactly this *sparsity*. It densifies the attribute space via plausible synthetic images that venture into potentially undersampled regions of the attribute spectra. Our approach does not expect to get “something for nothing”. Indeed, the synthesized examples are still annotated by humans. The idea is to expose the learner to realistic images that are critical for fine-grained visual learning yet difficult to attain in traditional data collection pipelines.

6.2 Evaluation

We conduct fine-grained visual comparison experiments to validate the benefit of our dense supervision idea, for both rankers described in Section 6.1.3.

6.2.1 Experimental Setup

Datasets Our experiments rely on the following existing and newly collected datasets.

- **Zap50K+New Lexicon:** Our improved UT-Zap50K dataset from Section 3.2.

- **Zap50K-Synth:** A new synthetic shoe dataset with pairwise labels on the new 10-attribute lexicon. We train the generative model using a subset of UT-Zap50K and a superset of the above attributes. We generate 1,000 identities and each one is used to sample both an intra- and inter-identity pair, yielding $\sim 2,000$ pair labels per attribute. The synthetic images are 64×64 pixels.
- **LFW-10:** Original LFW-10 dataset as described in Section 3.3.
- **PFSmile:** A small set of faces images also described in Section 3.3, which is exclusively used for evaluation on the face images.
- **LFW-Synth:** A new synthetic face dataset with pairwise labels on the attribute *smiling*. We train the generative model on a subset of LFW images and the 73 attributes from [63, 118]. We generate 2,000 identities and sample a total of 4,000 intra pairs and 1,000 inter pairs. The synthetic images are 35×35 pixels, after zooming to a tight bounding box around the face region.

Implementation Details For synthesis, we use the code shared by the authors for the Attr2Img system [118], with all default parameters. Since we inherit the generator’s 64×64 output image resolution, for apples-to-apples comparison, we downsize real images to match the resolution of the synthetic ones. Early experiments showed that a mix of inter and intra-identity pairs was most effective, so we use a 50-50 mix in all experiments. For RankSVM, we use Gist [110] and 30-bin Lab color histograms as the image features ϕ , following [84, 125]⁴, and validate K per method on held-out data. For DeepSTN, we use training parameters provided in [107] per dataset. The images used to train the generative model, to train the ranking functions, and to evaluate (test set) are kept strictly disjoint.

Baselines We compare the following methods:

- **Real:** Training pool consists of only real image pairs, labeled by human annotators.
- **Jitter:** Uses the same real training pairs, but augments them with pairs using traditional low-level jitter. Each real image is jittered following parameters in [25]

⁴Pretrained CNN features with RankSVM proved inferior.



Figure 6.5: Sample images generated through traditional low-level jitter. Each image is seeded from a real image in the training set and the modifications are attribute-independent.

in a combination of five changes: translation, scaling, rotation, contrast, and color (Fig. 6.5). A jittered pair inherits the corresponding real pair’s label.

- **SemJitter:** Training pool consists of only *half* of Real’s pairs, with the other half replaced with our dense synthetic image pairs, manually verified by annotators.
- **SemJitter-Auto:** Training pool consists of all real image pairs and our automatically supervised synthetic image pairs, where *noisy* pairwise supervision is obtained (for free) based on the absolute attribute strength used to generate the respective images.
- **Classifier:** Predicts the attribute scores directly using the posterior $R_{\mathcal{A}}(\phi(\mathbf{x})) = p(\mathcal{A}|\mathbf{x})$ obtained from a binary classifier trained with the same images that train the image generator.
- **Real+:** Augments Real with additional pseudo real image pairs. The image generator [118] requires attribute strength values on its training images, which are obtained from outputs of an attribute classifier [63]. The Real+ baseline trains using the same real pairs used above, plus pseudo pairs of the equal size bootstrapped from those strength values on individual images.

We stress that our semantic jittering methods use the same amount of human-annotated pairs as the Real and Jitter baselines.

6.2.2 Fashion Images of Shoes

Fashion product images offer a great testbed for fine-grained comparisons. This experiment uses UT-Zap50K for real training and testing pairs, and Zap50K-Synth for

Table 6.1: Results on Zap50K for the new lexicon of 10 attributes most frequently used to distinguish fine-grained differences between shoe images. We experiment with two kinds of base training models: (top) local learning (RankSVM) [125] and (bottom) localized ranking (DeepSTN) [107].

		Comfort	Casual	Simple	Sporty	Color	Durable	Support	Bold	Sleek	Open
Classifier		72.69	79.32	82.20	81.21	17.87	80.05	78.62	18.35	17.60	27.15
RankSVM	Real	84.03	86.11	86.89	87.27	83.84	85.15	87.75	83.71	86.06	84.41
	Real+	82.41	87.04	86.18	87.58	84.79	84.69	87.75	81.44	88.02	81.18
	Jitter	84.49	87.35	88.52	83.36	85.36	86.77	86.86	85.36	86.31	82.53
	SemJitter	85.02	88.89	85.56	89.95	87.43	84.32	87.29	87.62	86.40	81.05
	SemJitter-Auto	84.72	87.35	87.59	86.06	85.74	86.78	83.74	85.36	86.55	83.87
DeepSTN	Real	84.95	87.04	89.46	88.79	94.30	83.29	85.75	87.42	85.82	84.68
	Real+	81.25	87.65	86.18	87.88	90.68	83.29	85.52	87.84	86.31	82.53
	Jitter	81.94	87.96	86.89	87.58	93.73	85.38	85.75	89.07	83.86	80.65
	SemJitter	82.18	89.81	89.70	90.30	93.73	87.24	85.52	89.28	86.55	82.26
	SemJitter-Auto	87.27	88.89	88.76	90.00	95.44	88.86	87.75	87.63	86.80	86.29

synthetic training pairs. There are 10 attributes total. Since the real train and test pairs come from the same dataset, this presents a challenge for our approach—can synthetic images, despite their inherent domain shift, still help the algorithm learn a more reliable model?

Table 6.1 shows the results. The Classifier baseline underperforms both rankers, confirming that the generator’s initial representation of attribute strengths is insufficient.

Under the local RankSVM model, our approach outperforms the baselines in most attributes. Augmenting with traditional low-level jitter also provides a slight boost in performance, but not as much as ours. Figure 6.5 shows examples of low-level jittered images. Looking at the composition of the local neighbors, we see that about 85% of the selected local neighbors are our synthetic pairs (15% real) while only 55% are jittered pairs (45% real). Thus, our synthetic pairs do indeed heavily influence the learning of the ranking models. Figure 6.4 shows examples of nearest neighbor image pairs retrieved for sample test pairs. The examples illustrate how 1) the synthetic images densify the supervision, providing perceptually closer instances for training, and 2) both real and synthetic image pairs play an important role in training. We conclude that semantic jitter densifies the space more effectively than low-level jitter. In Chapter 7, I provide results to analyze more deeply how the space is densified and to what extent the generated samples follow the distribution of the real images.

Under the DeepSTN model (Table 6.1), our approach outperforms the baselines in all attributes. Interestingly, SemJitter-Auto often outperforms SemJitter here. We believe the higher capacity of the DeepSTN model can better leverage the noisy auto-

Table 6.2: Results on UT-Zap50K-1 (coarse pairs) and UT-Zap50K-2 (fine-grained pairs) vs. prior methods. Refer to Section 3.1 for the definitions of the coarse and the fine-grained pairs. All methods are trained and tested on 64×64 images for an apples-to-apples comparison. All experimental setup details are kept the same except for the addition of dense synthetic pairs to the training pool for our approach.

		Open	Sporty	Comfort
Zap50K-1	RelAttr [84]	88.33	89.33	91.33
	FG-LP [125]	90.67	91.33	93.67
	DeepSTN [107]	93.00	93.67	94.33
	SemJitter-Auto (Ours)	95.00	96.33	95.00
Zap50K-2	RelAttr [84]	60.36	65.65	62.82
	FG-LP [125]	69.36	66.39	63.84
	DeepSTN [107]	70.73	67.49	66.09
	SemJitter-Auto (Ours)	72.18	68.70	67.72

labeled pairs, compared to the RankSVM model, which more often benefits from the human-verification step. As one would expect, we notice that SemJitter-Auto does best for attributes where the inferred labels agree most often with human provided labels. This is an exciting outcome; our model has potential to generate useful training data with “free” supervision. Low-level jitter on the other hand has limited benefit, even detrimental in some cases. Furthermore, the number of synthetic pairs used correlates positively with performance, e.g., halving the number of synthetic pairs to SemJitter-Auto decreases accuracy by 4 points on average.

For both ranking models, our approach outperforms the Real baseline. This shows that simply collecting more annotations on real images is not enough: “Real” uses twice as many real training pairs as our method, yet is consistently less accurate. The finding holds even when we augment Real with [118]’s instance labels (Real+). Both baselines suffer from the *sparsity issue*, lacking the fine-grained comparisons needed to train a stronger model.

Overall, our gains are significant, considering they are achieved without any changes to the underlying ranking models, the features, or the experimental setup.

Comparison to Prior Relative Attribute Results Next, we take the best model from above (DeepSTN+SemJitter-Auto), and compare its results to several existing methods. While authors have reported accuracies on this dataset, as-is comparisons to our model would not be apples-to-apples: due to the limits of image synthesis at the

Table 6.3: Results on PFSmile dataset. The Classifier baseline is independent of the data composition, and therefore, takes up its own row.

Smiling	Real	Real+	Jitter	SemJitter	SemJitter-Auto
Classifier	-	-	-	62.35	-
RankSVM	69.29	68.95	74.29	73.88	75.00
DeepSTN	81.52	80.84	80.09	85.78	84.36

time of our implementation, we work with low resolution data (64×64) whereas prior attribute work uses full sized 150×100 [107, 108, 117]. Therefore, we use the authors’ code to re-train existing methods from scratch with the same smaller real images we use. In particular, we train 1) Relative attributes (**RelAttr**) [84]; 2) Fine-grained local learning (**FG-LP**) [125]; and 3) End-to-end localization and ranking (**DeepSTN**). We compare them on UT-Zap50K.⁵ To avoid an unfair advantage, we only use relevant real image pairs from the original UT-Zap50K dataset (Sec. 3.1), as opposed to those from our improved UT-Zap50K dataset (Sec. 3.2).

Tables 6.2 shows the results. Our approach does best, improving the state-of-the-art DeepSTN even for the difficult fine-grained pairs on UT-Zap50K-2 where attention to subtle details is necessary.

6.2.3 Human Faces

Next we consider the face domain. This experiment uses LFW-10 for real training pairs, LFW-Synth for synthetic training pairs, and PFSmile for real testing pairs (see Sec. 3.3). Since PFSmile only contains image pairs of the same individual, the comparison task is fine-grained by design. Here we have an additional domain shift, as the real train and test images are from different datasets with somewhat different properties.

Table 6.3 shows the results. Consistent with above, our approach outperforms all baselines. Even without human verification of our synthetic pairs (SemJitter-Auto), our method secures a decent gain over the Real baseline: 75.00% vs. 69.29% and 84.36% vs. 81.52%. That amounts to a relative gain of 8% and 3.5%, respectively. The Classifier posterior baseline underperforms the rankers. Our semantic jitter strongly outperforms traditional low-level jitter for the DeepSTN rankers, with a 6 point accuracy boost.

⁵We test all attributes that overlap between UT-Zap50K and Zap50K-Synth, namely “open”, “sporty”, and “comfort”.

6.3 Discussion

In this chapter, I proposed a new data augmentation approach to overcome the sparsity of supervision issue in fine-grained comparison learning. The proposed approach involves the generation and the addition of synthetic image pairs into the existing pool of real training image pairs. These synthetic pairs exhibit slight modifications of individual attributes, thus densifying the image space to better illustrate fine-grained differences. A key observation here is that sample *density* is distinct from sample *quantity*. Moreover, as shown through evaluation over two attribute domains, the improvements achieved through densifying supervision are independent of the prediction models used, further extending the generalizability of the proposed approach.

However, while the results are positive, there are various aspects of the approach that could be improved. Firstly, there is no clear winner between SemJitter and SemJitter-Auto, where each seems to perform better on certain attributes and not the others. Identifying the cause of this difference could help us determine when to elicit human annotators and when not to. Secondly, we are only modifying one single attribute when sampling for intra-identity pairs, which is not always realistic as the attributes are oftentimes correlated. Thirdly, we do not have a way to quantify the density of the attribute space, to validate that we are indeed densifying the training space. Finally, the current pipeline lacks a systematic way of sampling for the *identities*, or rather, for the synthetic image *pairs* in general. As we will see in Chapter 7, the next step in our data augmentation approach is to move away from a passive generation of images based on heuristics to a more active generation of the synthetic image pairs directly.

Chapter 7

Active Training Image Creation

Continuing with the issue of the sparsity of supervision in fine-grained comparisons, I now explore a holistic approach to synthesizing novel image pairs optimized for training ranking models. In the previous chapter, we first generate the synthetic images through individual *identities* and then form the supervision pairs based on automated sampling heuristics (Sec. 6.1.2). In this work, I propose to generate the image pairs directly in an adversarial manner, through active image generation. Unlike traditional active learning methods where informative instances—which are selected from a pool of manually curated unlabeled images—are prioritized for labeling, we design a system that directly synthesizes image pairs that would confuse the current ranking model for labeling. We refer to this approach as *ATTIC*, for **AcTive Training Image Creation**. The main idea is to jointly learn the target visual task while also learning to generate novel realistic image pairs that, once manually labeled, will benefit that task.

To this end, I propose an end-to-end framework for attribute-based image comparison, which serves as a continuation of our semantic jittering approach from Chapter 6. The adversarial aspect of the model aims to avoid the “streetlight effect” of traditional pool-based active learning. The “streetlight effect” refers to a type of observational bias where one searches for something only where it is the easiest to look, such as looking for a lost key only around a streetlight where it is well-lit. In our case, this is analogous to an active learning approach that only looks for new images to annotate from a pre-defined pool of images. Therefore, rather than limit training to manually curated real images, *ATTIC* synthesizes image pairs that will be difficult for the ranker as it has been trained thus far. See Figure 7.1.

In this chapter, we first describe the key modules in our *ATTIC* network (Sec. 7.1), followed by our training procedure and the active image creation active loop (Sec. 7.1.4). We validate our results once again on the fine-grained shoes and faces datasets, with multiple attributes for both datasets (Sec.7.2).

The work in this chapter also appears in arXiv [129].

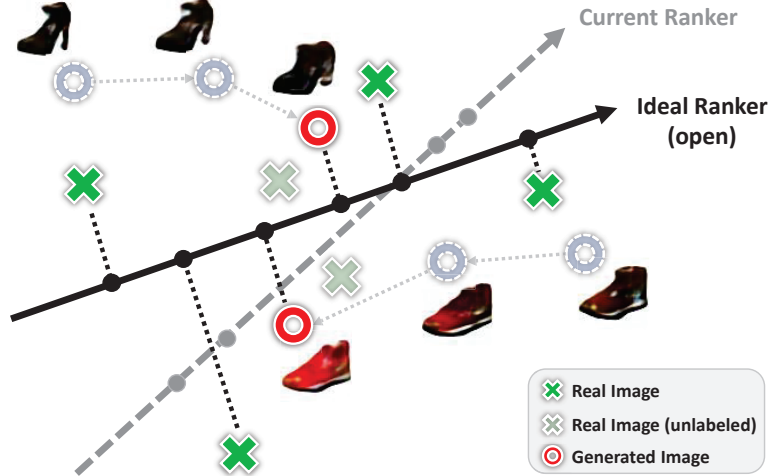


Figure 7.1: Schematic overview of main idea. Real images (green \times 's) are used to train a deep ranking function for the attribute (e.g., the *openness* attribute for shoes). The pool of real images consists of those that are labeled (dark \times 's) and those that are unlabeled (faded \times 's). Even with all the real images labeled, the ideal ranking function may be inadequately learned. Rather than select other manually curated images for labeling (faded green \times 's), ATTIC directly generates useful synthetic training images (red \circ 's) through an adversarial learning process. The three shoes along each path of circles represent how ATTIC iteratively evolves the control parameters to obtain the final synthetic image pairs.

7.1 Approach

Let \mathcal{P}_A be an initial set of real training image pairs used to initialize the ranker. Just like in the previous chapter, our goal is to improve that ranker by creating synthetic training image pairs \mathcal{S}_A , to form a hybrid training set $\{\mathcal{P}_A \cup \mathcal{S}_A\}$.

Our proposed end-to-end ATTIC framework consists of three distinct components (Fig. 7.2): the ranker module, the generator module, and the control module. Our model performs end-to-end adversarial learning between the ranker and the control modules. The ranker tries to produce accurate attribute comparisons, while the control module tries to produce *control parameters*—latent image parameters—that will generate difficult image pairs to confuse the current ranker. By asking human annotators to label those confusing pairs, the ranker is actively improved. Compared to my semantic jitter approach from Chapter 6, our novelty in this work comes from the addition of the control module, as well as the formation of an end-to-end adversarial network using all three modules. We next discuss the individual modules.

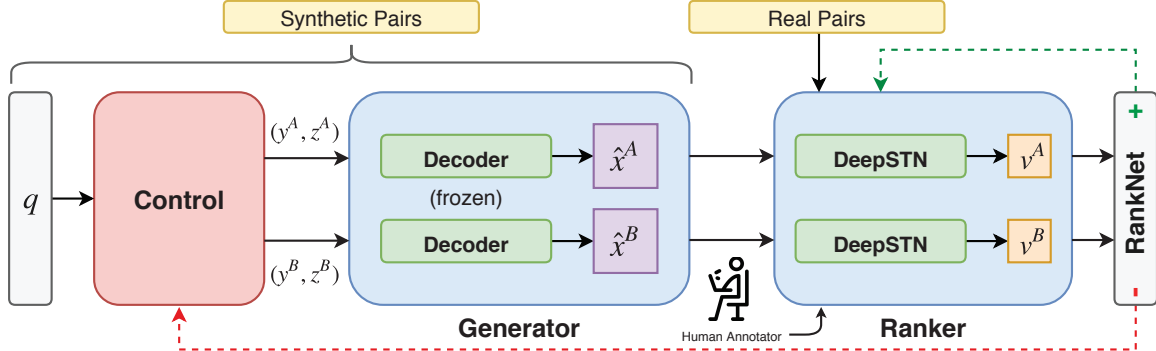


Figure 7.2: Architecture of our proposed end-to-end approach consisting of three primary modules. The control module first converts the random input q into control parameters $\{(y^A, z^A), (y^B, z^B)\}$. Its architecture is detailed further in Figure 7.3. The generator module then generates a pair of synthetic images (\hat{x}^A, \hat{x}^B) using these control parameters. The ranker module finally uses the generated synthetic images (once manually labeled) and the real training images to train the ranking model, outputting their corresponding attribute strength (v^A, v^B) . During training, the ranking loss using the RankNet objective is fed back into the ranker (green dotted line), while the negative ranking loss from the same objective is fed back into the control module (red dotted line). Note that the decoders within the generator are pre-trained and their parameters are kept frozen throughout training.

7.1.1 Ranking Module

For the ranking module in ATTIC, we once again employ the state-of-the-art deep DeepSTN with a spatial transformer network approach [107] detailed in Section 6.1.3. As before, RankNet handles pairwise outputs in a single differentiable layer using cross-entropy loss. The rank estimates (v_i, v_j) for images $(\mathbf{x}_i, \mathbf{x}_j)$ are mapped to a pairwise posterior probability using a logistic function

$$p_{ij} = \frac{1}{1 + e^{-(v_i - v_j)}}, \quad (7.1)$$

and the ranking loss is:

$$\mathcal{L}_{rank} = -\log(p_{ij}). \quad (7.2)$$

The ranking loss here is a simplified form of Equation 4.3 where only ordered pairs \mathcal{P}_o are used (i.e., $t_{ij} = 1$),

7.1.2 Generator Module

For the generator module, we adopt the same attribute-conditioned image generator, Attr2Image [118, 119], that was introduced in Section 6.1.1. Instead of generating synthetic image pairs offline based on sampling from hallucinated *identities*, we connect

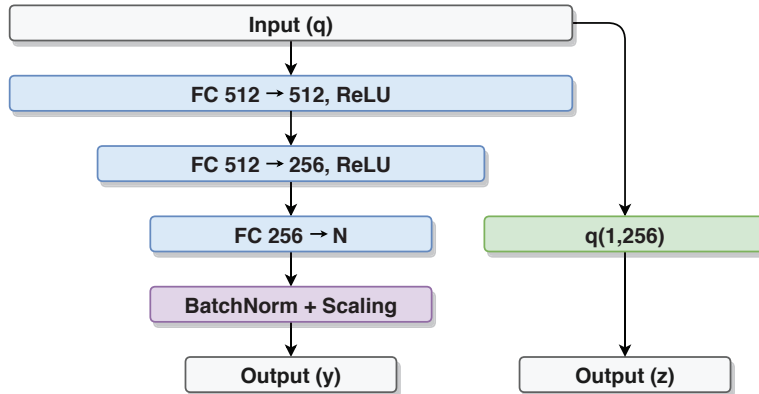


Figure 7.3: Architecture of the control module. The model above outputs a single set of control parameters (\mathbf{y}, \mathbf{z}) . Since we generate the synthetic images in pairs, we duplicate the architecture.

the generator outputs directly to the inputs of the ranker module in a Siamese manner. Synthetic image pairs are then generated and modified on-the-fly throughout training.

The *attribute-conditioned* aspect of this generator allows us to iteratively refine the generated images in a semantically smooth manner, as we adversarially update its inputs (\mathbf{y}, \mathbf{z}) with the control module defined next. We pre-train the generator using $\{(\mathbf{x}_i, \mathbf{y}_i)\}$, a disjoint set of training images labeled by their N_a attribute strength labels. Subsequently, we take only the decoder part of the model and use it as our generator (see Fig. 7.2). We freeze all parameters in the generator during active image creation, since the mapping from latent parameters to pixels is independent of the rank and control learning.

7.1.3 Control Module

As defined thus far, linking together the ranker and generator would aimlessly feed new image sample pairs to the ranker. Next we define our control module and explain how it learns to feed pairs of intelligently chosen latent parameters to the generator for improving the ranker.

The control module is a neural network that precedes the generator (see Figure 7.2, left). Its input is a random seed $\mathbf{q} \in \mathbb{R}^Q$, sampled from a multivariate Gaussian. Its output is a pair of *control parameters* $\{(\mathbf{y}^A, \mathbf{z}^A), (\mathbf{y}^B, \mathbf{z}^B)\}$ for synthetic image generation. Figure 7.3 shows the control architecture. It is duplicated to create two branches feeding to the generator and then the Siamese network in the DeepSTN ranker.

The attribute control variable \mathbf{y} is formed by passing \mathbf{q} through a few fully-

connected layers, followed by a BatchNorm layer with scaling. In particular, for the scaling we obtain the scaling parameters from the mean and the standard deviation of the attribute strengths observed from the real training images, then apply them to the normalized $\mathcal{N}(0, 1)$ outputs from the BatchNorm layer. The scaling ensures that the attribute strengths are bounded within a range appropriate for the pre-trained generator.

The latent feature control variable \mathbf{z} , which captures all the non-attribute properties (e.g., pose, illumination), is sampled from a Gaussian. We simply use half of the entries from \mathbf{q} for \mathbf{z}^A and \mathbf{z}^B , respectively. This Gaussian sample agrees with the original image generator’s prior $p(\mathbf{z})$ [118, 119].

7.1.4 Training and Active Image Creation

Given the three modules, we connect them in sequence to form our active learning network. The generator and the ranker modules are duplicated for both branches to account for two images in each training pair. The decoders in the generator module are pre-trained and their parameters are kept frozen. During training, we optimize the RankNet loss for the ranker module, while at the same time optimizing the negative RankNet loss for the control module:

$$\mathcal{L}_{control} = -\mathcal{L}_{rank}, \tag{7.3}$$

creating the adversarial effect. The control module thus learns to produce parameters that generate image pairs that are difficult for the ranker to predict. This instills an adversarial effect where the control module and the ranker module are competing against each other iteratively during training. The learning terminates when the ranker converges or reaches a certain threshold of training iterations.

To generate a batch¹ of synthetic image pairs $\mathcal{S}_A = \{\hat{\mathbf{x}}^A, \hat{\mathbf{x}}^B\}_{i=1}^T$, we sample T vectors \mathbf{q} and push them through the control and generator. Then the batch is labeled by annotators, who judge which image shows the attribute more, and the resulting pairs accepted by annotators as valid are added to the hybrid training set $\{\mathcal{P}_A \cup \mathcal{S}_A\}$. Figure 7.4 shows examples of the *progression* of some synthetic image pairs during the training iterations. As we can see, ATTIC captures the joint interaction between the

¹We use “batch” here in the active learning sense: a *batch* of additional examples are manually labeled then used to update the predictive model. This is not to be confused with (*mini*)-*batches* for training the neural networks.

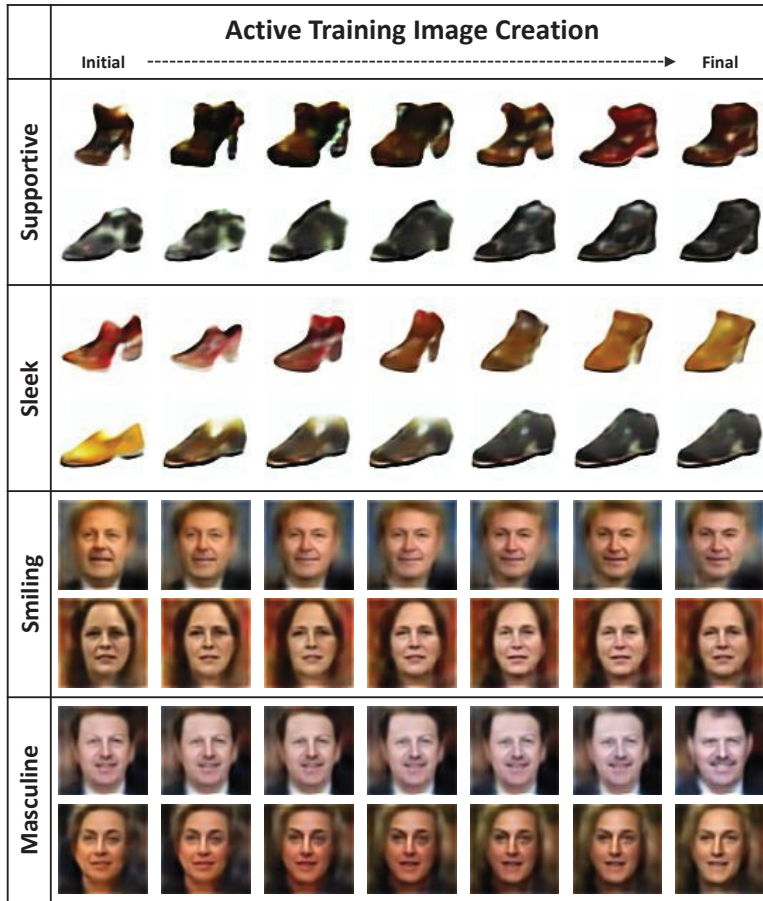


Figure 7.4: Visualization of the progression of some synthetic image pairs (\hat{x}^A, \hat{x}^B) during training. Our model learns patterns between all the attributes, modifying multiple attributes simultaneously. For example, while modifying the face images for the attribute *masculine* (last row), our model learned to change the attribute *smiling* as well.

attributes and modifies each pair of images simultaneously in order to best confuse the current ranking model.

The primary novelty of this approach comes from the generation of synthetic image pairs through active query synthesis. From an active learning perspective, instead of selecting more real image pairs to be labeled based on existing pool-based strategies, our approach aims to directly generate the most beneficial synthetic image pairs (Fig. 7.1). Furthermore, instead of sampling $(\mathbf{y}_i, \mathbf{z}_i)$ using a heuristic when generating the synthetic image pairs, as proposed in the previous chapter, ATTIC automates this selection in a data-driven manner.

7.2 Experiments

To validate our active generation approach, we once again explore the fine-grained shoe and face domains.

7.2.1 Experimental Setup

Datasets: For each domain, our method uses real images to initialize training and then creates its own synthetic training images. Note that the synthetic images are labeled by human annotators (in most cases) before they are used to augment the training set.

- **Catalog Shoe Images:** We use the improved **UT-Zappos50K** dataset with the fine-grained attributes from Section 3.2. There are 10 attributes (*comfort, casual, simple, sporty, colorful, durable, supportive, bold, sleek, and open*), each with about 4,000 labeled pairs.
- **Human Face Images:** We use the **LFW** dataset and the **LFW-10** dataset from Section 3.3. We use the 8 attributes (*bald, dark hair, big eyes, masculine, mouth open, smiling, visible forehead, and young*) in the intersection of these two datasets. For the real image pairs, there are about 600 labeled pairs per attribute from LFW-10.

All images for all methods are resized to 64×64 pixels to match the output resolution of the image generator. We collect annotations for our method’s automatically generated synthetic training pairs using mTurk, as we did with the real training pairs; we obtain five worker responses per label and take the majority vote (see Sec. 3.2 and 3.3). Workers are free to vote for discarding a pair if they find it illegible, which happened for just 17% of the generated pairs.

Implementation Details: During training, we validate all hyperparameters (such as the learning rate, the learning rate decay, and the weight decay) on a separate validation set. We run all experiments (including individual batches) to convergence or to a maximum of 250 and 100 epochs for shoes and faces, respectively. We monitor the ranking loss on the validation set throughout training to avoid overfitting.

For the individual modules, implementation details are as follows. Ranker: We pre-train the DeepSTN ranking network without the global image channel using only

the real image pairs (see [107] for details on the two rounds of training). **Generator:** We use the code provided by the authors of Attr2Image [118] with all default parameters. We pre-train the image generators using a disjoint set of real images (38,000 and 11,000 images for shoes and faces, respectively) that do not have any associated relative labels. We use the trained decoder in ATTIC while keeping the parameters constant throughout end-to-end training for the ranker and control (i.e., learning rate of zero on decoder). **Control:** We initialize the layers using ReLU initialization [41]. The learning rate decays such that as learning goes on, the changes to \mathbf{y} , \mathbf{z} become smaller.

Baselines: We consider the following baselines.

- **Real:** Standard approach which trains with only real labeled image pairs.
- **Real+:** Slight modification that adds real image pairs with their pseudo labels to Real. The purpose of this baseline is to ensure that our advantage is not due to our network’s access to the attribute-strength labeled images that the image generator module requires for training.
- **Jitter:** The traditional data augmentation process where the real images are jittered through low-level geometric and photometric transformations. We follow the jitter protocol defined in [25], which includes translation, scaling, rotation, contrast, and color. The jittered image pairs retain the corresponding real pairs’ respective labels.
- **Semantic Jitter:** Our dense supervision approach from Chapter 6. For the synthetic shoe image pairs, we use the Zap50K-Synth dataset. For the synthetic face image pairs, we collect relative labels on 1,000 pairs per attribute.

Overall, the above baselines are designed in the exact same way as they are in Section 6.2.1. All methods use the same state-of-the-art ranking network for training and predictions, hence any differences in results will be attributable to the training data and augmentation strategy. Notice that we do not evaluate on the large-margin ranking function (RankSVM), since it cannot be integrated into our end-to-end pipeline.

Table 7.1: Accuracy for the 10 attributes in the Shoes dataset. Semantic Jitter is the approach proposed in Chapter 6. “Normal” means that the synthetic images generated by Semantic Jitter and ATTIC are labeled by human annotators. “Auto” means that an additional n *unlabeled* synthetic image pairs are added for all methods except Real; those images adopt their inferred attribute labels. In all cases, all methods use exactly n total labels. The row for the Real baseline is repeated for Normal and Auto for easier comparison purposes.

		Comfort	Casual	Simple	Sporty	Colorful	Durable	Supportive	Bold	Sleek	Open
Normal	Real [107]	84.26	88.58	88.99	88.18	94.10	82.83	83.96	88.25	84.35	83.87
	Real+	81.71	87.96	87.12	87.58	91.05	82.60	84.41	87.63	85.82	83.87
	Jitter	79.17	88.88	85.48	85.76	92.00	79.81	80.85	87.63	82.89	78.50
	SemJitter	83.10	89.20	88.76	88.49	94.10	82.37	85.08	89.07	86.31	82.26
	ATTIC	83.80	89.51	89.23	88.18	94.10	85.85	86.41	88.04	87.78	83.07
Auto	Real [107]	84.26	88.58	88.99	88.18	94.10	82.83	83.96	88.25	84.35	83.87
	Real+	81.71	87.96	87.12	87.58	91.05	82.60	84.41	87.63	85.82	83.87
	Jitter	82.87	87.96	86.65	87.58	93.91	80.51	84.63	88.66	83.37	79.84
	SemJitter	84.72	88.89	89.70	89.39	94.29	83.99	86.41	89.07	85.82	83.87
	ATTIC	87.04	89.20	91.57	91.21	94.48	87.94	87.31	89.90	86.31	85.75

7.2.2 Pairwise Comparisons using ATTIC

First, we validate our hypothesis from the beginning of this chapter, that our approach could overcome the “streetlight effect” of traditional pool-based active learning. We compare the data augmentation baselines and ATTIC to the Real baselines, where all methods are given the exact same amount of total manual annotations. The Real, Real+, and Jitter baselines use all n available real labeled image pairs. Semantic Jitter and ATTIC use *half* of the real labeled image pairs ($\frac{n}{2}$), then augment those pairs with $\frac{n}{2}$ manually labeled synthetic image pairs that they generate.

Tables 7.1 and 7.2 (Normal) show the results for the shoe and face datasets, respectively. Though using exactly the same amount of manual labels as the Real baseline, our method nearly always outperforms it. This shows that simply having more real image pairs labeled is not always enough; our generated samples improve the training across the variety of attributes in ways the existing real image pairs could not. In addition, we see from Real+ that the image generator training images have only a marginal (and sometimes negative) effect on the baseline’s results. This indicates that both Real and Real+ suffer from the same sparsity issue, as the images are taken from similar pool of real images. The addition of similarly distributed (real) images lacks the fine-grained details needed to train a stronger model. This is consistent with our observation from Section 6.2. Furthermore, our approach also outperforms (or matches) Semantic Jitter in 8 out of 10 shoe attributes and 6 out of 8 face attributes, with gains of just over 3% in some cases. This demonstrates our key advantage over Semantic Jitter, which is

Table 7.2: Accuracy for the 8 attributes in the Faces dataset. Format is the same as Table 7.1.

		Bald	DarkHair	BigEyes	Masculine	MouthOpen	Smiling	Forehead	Young
Normal	Real [107]	79.80	86.77	78.18	92.96	87.50	74.44	80.00	78.76
	Real+	81.82	86.03	80.00	92.96	86.67	75.94	81.21	79.28
	Jitter	80.81	85.29	76.36	88.73	77.50	74.44	81.05	77.20
	SemJitter	81.82	87.50	83.64	92.96	88.33	79.70	83.16	81.35
	ATTIC	84.85	88.24	85.46	95.78	79.17	81.96	84.21	80.31
Auto	Real [107]	79.80	86.77	78.18	92.96	87.50	74.44	80.00	78.76
	Real+	81.82	86.03	80.00	92.96	86.67	75.94	81.21	79.28
	Jitter	81.82	86.87	80.00	91.55	84.17	74.44	85.26	79.79
	SemJitter	82.83	88.24	81.82	94.37	85.00	75.19	83.16	79.28
	ATTIC	85.86	90.44	83.36	94.37	82.50	76.69	85.26	78.24

to actively adapt the generated images to best suit the learning of the model, as opposed to what looks the best to human eyes. Unlike Semantic Jitter, which modifies one attribute at a time, our approach can modify multiple attributes simultaneously in a dynamic manner, accounting for their dependencies.

In Tables 7.1 and 7.2 we also consider an “Auto” scenario where instead of adding the $\frac{n}{2}$ generated images with their manual annotations, we bootstrap from all n real labeled image pairs. Then, we generate another n synthetic images and—rather than get them labeled—simply adopt their inferred attribute comparison labels (equivalent to “SemJitter-Auto” from the previous chapter). In this case, the “ground truth” ordering for attribute j for generated images \hat{x}^A and \hat{x}^B is automatically determined by the magnitudes of their associated parameter values $\mathbf{y}^A(j)$ and $\mathbf{y}^B(j)$ output by the control module. As before, Jitter adopts the label of the source pair it jittered. Once again, all methods use the same number of labels.

Table 7.1 and 7.2 (Auto) show the results. Our model performs even a bit better in this setting, suggesting that the inferred labels are often accurate, and the extra volume of “free” training pairs is helpful. We outperform (or match) Semantic Jitter in all 10 shoe attributes and 6 out of 8 face attributes. Jitter gets a slight performance boost sometimes, but can even be detrimental on these datasets.

While our method performs well overall, for a couple of attributes (i.e, *mouth-open*, *young*) we underperform both Real and Semantic Jitter. Upon inspection, we find our weaker performance there is due to deficiency in the image generators. While both of our densifying approaches would suffer in such a scenario, ATTIC is more susceptible to generator errors due to its adversarial nature.²

²Also, the minor variations between Table 6.1 and Table 7.1 are due to subtle implementation differences between

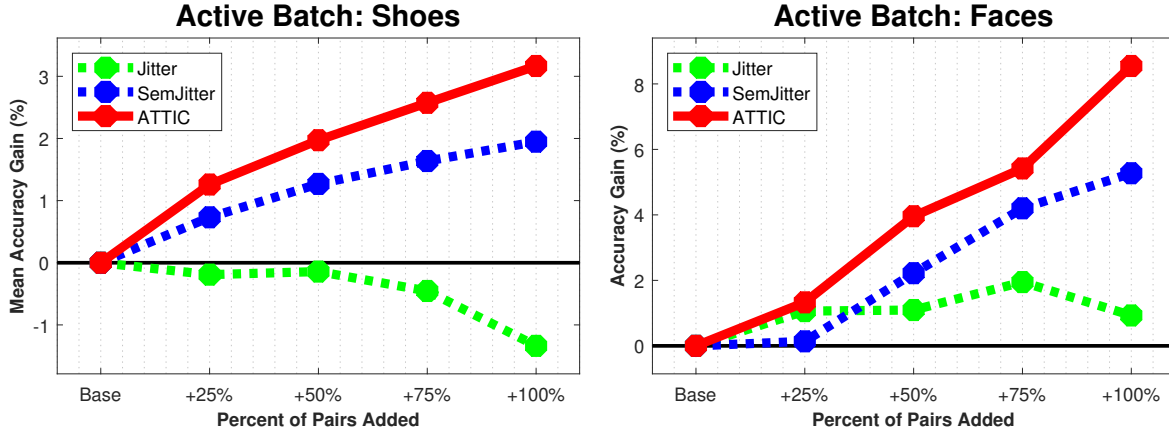


Figure 7.5: Active learning curves for the shoe (left) and face (right) datasets. We show the average gain over the Real baseline after each batch of additional generated image pairs. ATTIC nearly doubles the gain achieved by Semantic Jitter for both domains.

7.2.3 Active vs. Passive Training Image Generation

Next we examine more closely ATTIC’s active learning behavior. In this scenario, we suppose the methods have exhausted all available real training data (i.e., we use all n real labeled image pairs to initialize the model), and our goal is to augment this set. We generate the synthetic (labeled) image pairs in batches (again, not to be confused with the mini-batches when training neural networks). After each batch, we have them annotated, update the ranker’s training set, and re-evaluate it on the test set. The weights of the control module are carried over from batch to batch, while the ranker module restarts at its pre-trained state at the beginning of each batch.

Figure 7.5 shows the results for both datasets. We plot active learning curves to show the accuracy improvements as a function of annotator effort—steeper curves are better, as they mean the system gets more accurate with less manual labeling. We see the average gains of our approach over the Real baseline increase most sharply compared to the baselines. Our approach achieves a gain of over 3% and 8% for the two domains, respectively, which is almost double that of our Semantic Jitter approach. Jitter falls short once again, suggesting that traditional low-level jittering has limited impact in these fine-grained ranking tasks.

the two. For example, we added a validation set in this work to validate for the increased number of hyperparameters in our end-to-end model, whereas in Section 6.2 we simply used the existing hyperparameters from [107]. The inclusion of a validation set also changes the distribution of the original train/test splits, resulting in more potential variations in the final results. However, these minor variations do not affect the overall conclusion drawn in either work.

Table 7.3: Extension to Table 6.2 in Chapter 6 that includes our results for the same UT-Zappos50K splits. As before, all methods are trained and tested on 64×64 images for an apples-to-apples comparison.

		Open	Sporty	Comfort
Zap50K-1	RelAttr [84]	88.33	89.33	91.33
	FG-LP [125]	90.67	91.33	93.67
	DeepSTN [107]	93.00	93.67	94.33
	SemJitter [128]	95.00	96.33	95.00
	ATTIC	95.67	96.00	95.67
Zap50K-2	RelAttr [84]	60.36	65.65	62.82
	FG-LP [125]	69.36	66.39	63.84
	DeepSTN [107]	70.73	67.49	66.09
	SemJitter [128]	72.18	68.70	67.72
	ATTIC	71.68	69.62	68.64

7.2.4 Comparison to Previously Published Results

The experiments thus far demonstrate that our approach allows more accurate fine-grained predictions for the same amount of manual annotation effort, compared to both traditional training procedures with real images as well as existing jitter approaches. Next we present results for our approach alongside all available comparable reported results on the UT-Zap50K dataset.

Table 7.3 shows the results reported in Table 6.2 alongside our latest results from ATTIC using the same UT-Zappos50K train/test split. Following Section 6.2.2, for an apples-to-apples comparison, all methods are applied to the same 64×64 images. Our results use the method exactly as described above for the “Auto” scenario.

Our method outperforms all the existing methods for the majority of the attributes. Semantic Jitter outperforms ATTIC for *sporty* in the first test set and *open* in the second test set, indicating that those attributes were similarly well-served by that method’s heuristic choice for generated images. However, our automated method overall has the advantage.

7.2.5 Qualitative Analysis

As we have seen in the results above, the synthetic image pairs generated by our approach outperform those selected by the heuristic and passive selection processes of Semantic Jitter and Jitter in almost all scenarios. The advantage of our active generation

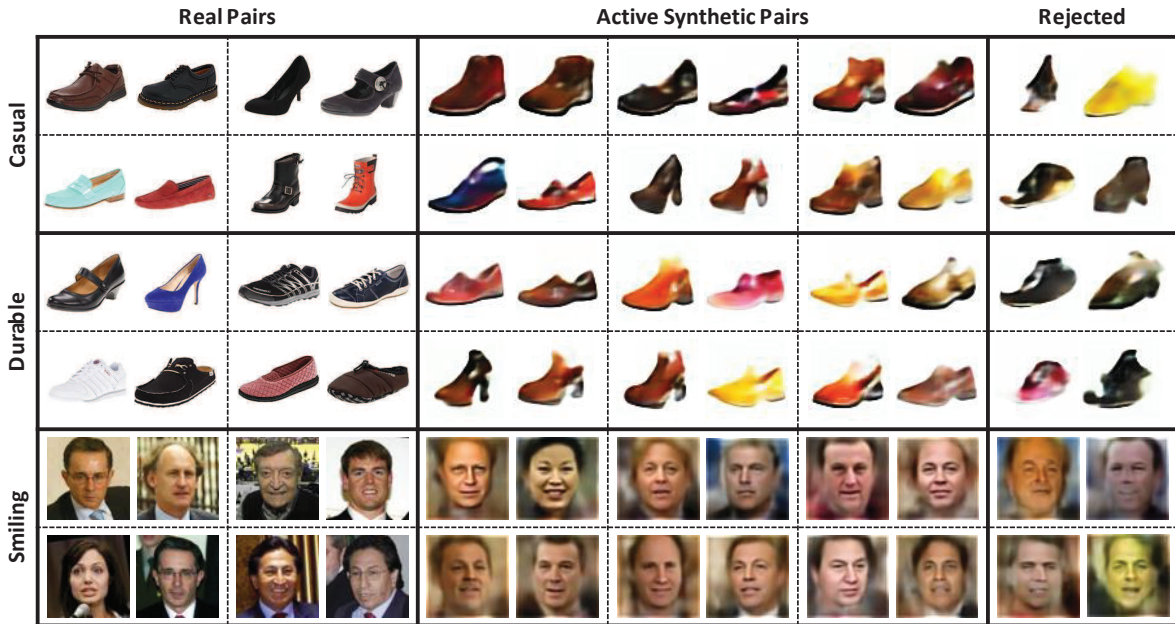


Figure 7.6: Sample training image pairs generated by ATTIC. “Harder” real pairs that are incorrectly predicted by the baseline model (left). Synthetic image pairs generated by our active approach (middle). Synthetic image pairs that are rejected by the human annotators during the labeling process as illegible (right).

approach is its ability to modify the generated image pairs in a way that is best for the learning of the model.

Figure 7.4 shows examples of how the synthetic images look between the first and the last epoch of the training. We can see that pairs generated by our approach demonstrate change in multiple attributes while still keeping the target attribute of comparison at the forefront. Furthermore, the final pairs selected for labeling also demonstrate subtler visual differences than the initial pairs, suggesting that our model has indeed learned to generate “harder” pairs.

Figure 7.6 compares the “harder” pairs generated by ATTIC to those from the real image pairs. Overall we see that the actively generated synthetic pairs tend to have fine-grained differences and/or offer visual diversity from the real training samples. The righthand side of the figure shows examples of generated pairs rejected by annotators as illegible, which occurs 17% of the time. The relative low rate of rejection is an encouraging sign for making active query synthesis viable for image labeling tasks.

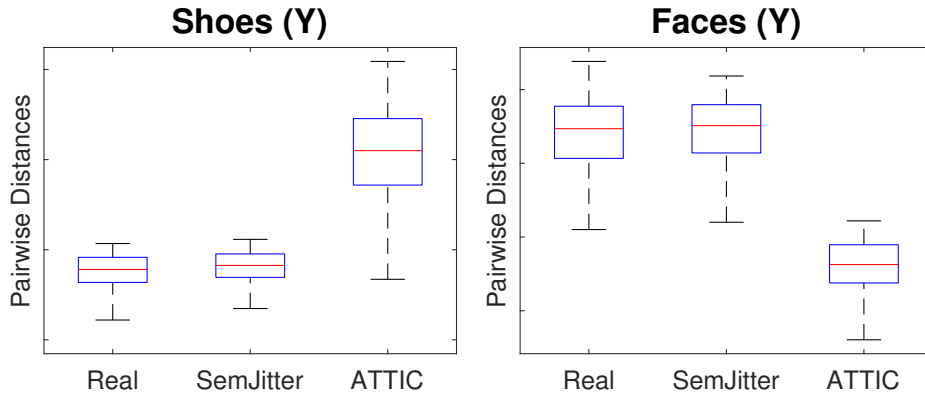


Figure 7.7: Box plots representing the intra-pair image distances of the various types of images used in our experiments. Each image is represented by its associated attribute values \mathbf{y} . The distances of all attributes from each dataset are combined for these computations.

7.2.6 Insights into Synthetic Images

Finally, having seen the performance benefits of using synthetic images for training ranking models, our last set of analysis aims to validate our original hypothesis from Chapter 6: Did our synthetic images *densify* the supervision in the attribute space? For our analysis, we represent each image using its associated attribute values \mathbf{y} .

First, we analyze the intra-pair image distances of the four types of images used: Real, Jitter, SemJitter, and ATTIC. A larger distance represents a greater difference in the respective feature space. Our automated sampling heuristics from Section 6.1.2 work under the assumption that shorter distances are better for learning fine-grained comparisons.

Figure 7.7 shows the box plot representation of these distances.³ Immediately, we see a clear contrast between the two image domains with our ATTIC image pairs. While the face images follow our hypothesis that pairs with lower distances are preferable, the shoe images actually exhibit the opposite preference. Our ATTIC shoe distances are not only larger than those from Real and SemJitter but also occupy a wider range, therefore, resulting in more diverse pairs. This makes sense for shoe images where the difference in the attributes are often highly correlated amongst one another, especially in relation to the 40 meta-data labels. Therefore, when automatically adjusting for one attribute, many of the attributes are most likely modified as well. Our ATTIC face distances on the other hand exhibit very subtle differences, representative of the fine-grained differences

³We do not have \mathbf{y} for Jitter baseline as they would be the same as their Real counterparts.

we are trying to identify on faces (e.g., minor changes in the size of the mouth for *smiling*). This stark difference between the generated image pairs in the two domains highlights the strength of our ATTIC approach, as it was able to adapt to each set of training data to generate the most beneficial set of images for the respective rankers. This adaptability is something that is missing from our semantic jittering approach.

Next, we attempt to visualize the distribution of these individual training images using 2D t-SNE [72] embeddings. Figure 7.8 shows the 2D t-SNE embedding space of images from Real, SemJitter, and the active batches of ATTIC. For the shoe domain, our ATTIC images are mostly positioned away from the existing real images, generating into regions in the feature space that are unoccupied. For the face domain on the other hand, our ATTIC images tend to be tightly clustered amongst one another, signaling fine-grained difference between the images. In addition, the clustering of the face images mostly happens around the center of the t-SNE map. This is likely due to the blurry backgrounds of the generated face images causing unintended uniformity during t-SNE optimization. Overall, these observations agree with the main outcomes in the box plots above.

Furthermore, we also generate another set of t-SNE grid visualization in Figures 7.9 and 7.10, where we display the actual images in their rough locations in the embedding space, while filling out the missing locations with its nearest neighbor in the attribute space. We color-code the images to represent their individual types.

Finally, a key observation to note from both analysis above is the difference between the synthetic images generated by our semantic jittering approach and by our ATTIC approach here. Even though both approaches use the exact same image generator (Attr2Image), Semantic Jitter densifies throughout the space, but sticks near the distribution of real images, for both datasets. In contrast, ATTIC ventures into new parts of the feature space for the shoe attributes, demonstrating its ability to adapt and adjust the learning to each specific set of training data.

7.3 Discussion

In this chapter, I proposed an approach for actively generating training image pairs that can benefit a fine-grained attribute ranker. ATTIC focuses attention on novel training image pairs that rapidly improve generalization—even after all available real images and

their labels are exhausted. It lets the system think “outside of the pool” in annotation collection, imagining its own training samples and querying human annotators for their labels. On two difficult datasets we showed that the approach offers real payoff in accuracy for distinguishing subtle attribute differences, with consistent improvements over existing data augmentation techniques that generate synthetic image pairs in a passive manner.

However, the success of ATTIC comes at a cost of the complexity of the end-to-end model. As with any adversarial system, finding the right balance between the learning of the ranker and the control modules is non-trivial. If the ranker module dominates, the generated pairs tend to converge to a single local minima, which represents the most “optimal” training pair. If the control module dominates, we get a “fooling network” effect where the generated images might be optimal for learning the ranker but visually incomprehensible. The use of the scaling layer in the control module alleviates this issue, but also limits the range of variability of the generated synthetic images in the process.

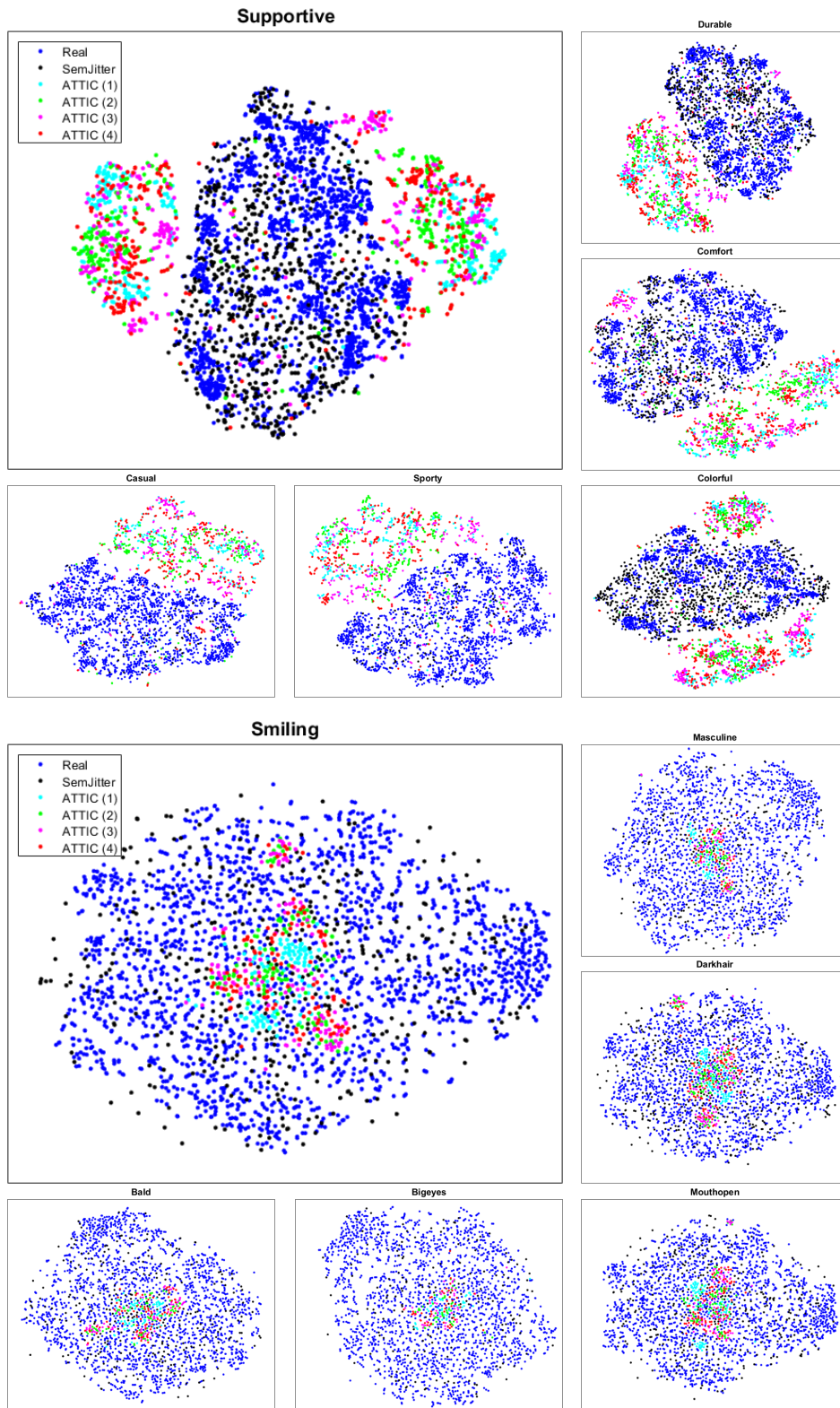


Figure 7.8: t-SNE visualization of various types of images, including Real, SemJitter, and ATTIC (active batches) for the shoe domain (top) and face domain (bottom). Each batch of our ATTIC approach is represented with a different color. See legend for their corresponding batch numbers. Best viewed on PDF.

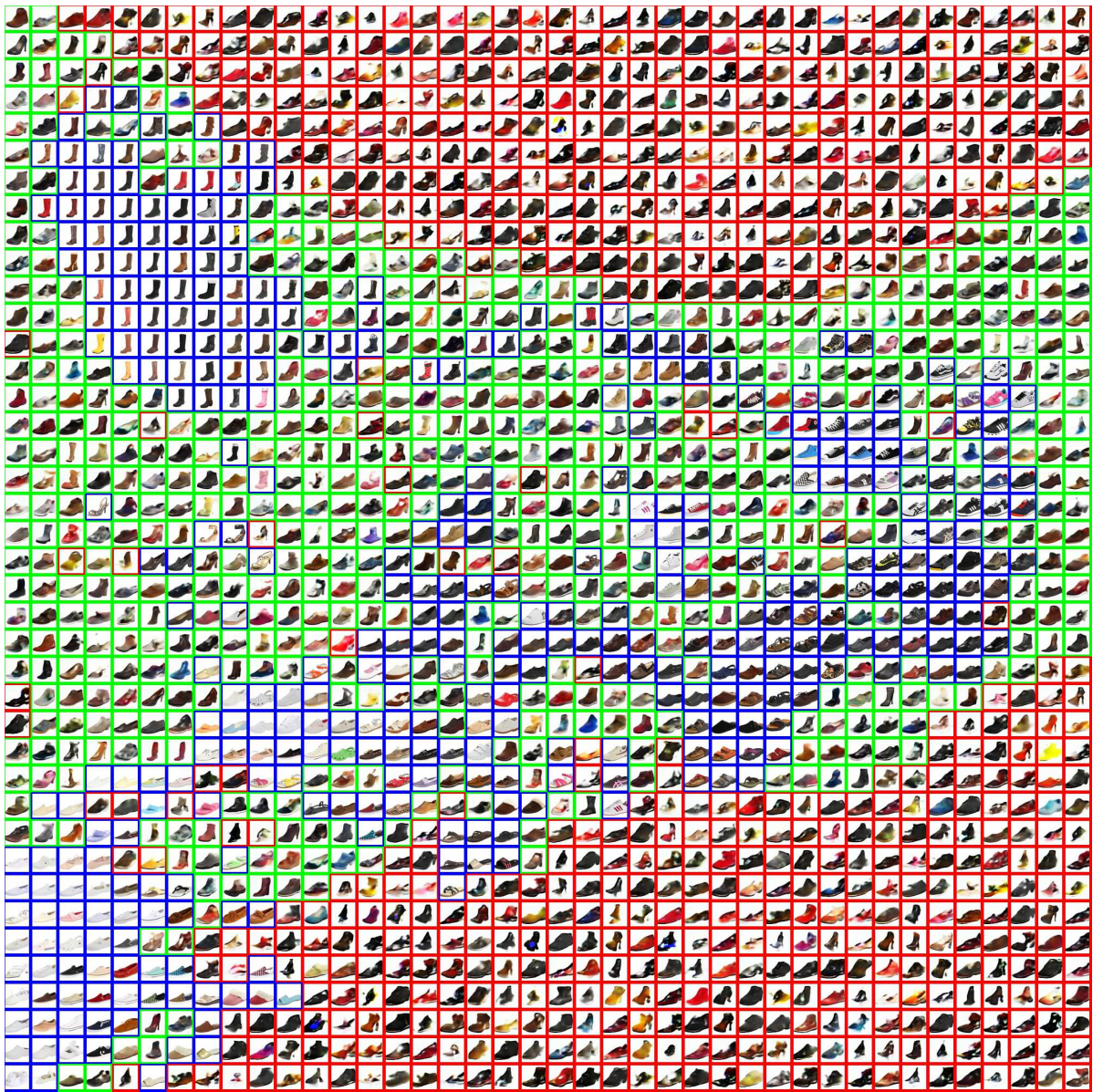


Figure 7.9: Alternate t-SNE grid visualization where every point in an embedding is filled with its nearest neighbor. We show here complementary visualizations to the main embedding from Figure 7.8, for the attributes *supportive* and *smiling* (in Fig. 7.10 below). The image border colors represent the type of images, with Real as blue, SemJitter as green, and ATTIC as red. Best viewed on PDF.

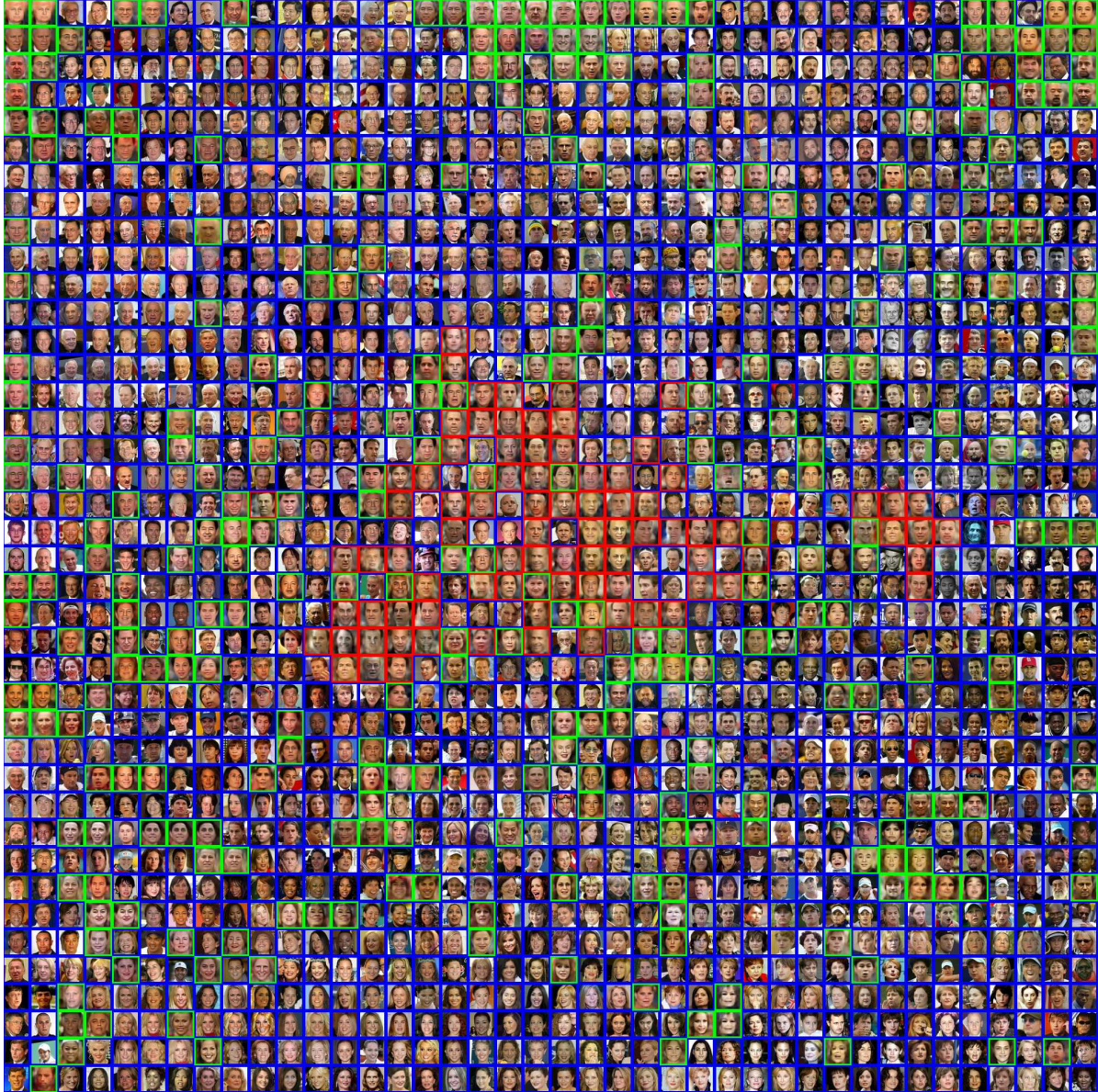


Figure 7.10: Alternate t-SNE grid visualization for the face attribute *smiling*. Best viewed on PDF.

Chapter 8

Conclusion

My thesis explores the task of fine-grained visual comparisons and introduces major improvements over two fronts. Whether it was by using algorithms that prioritized local neighbors or by directly generating the necessary neighboring images, we saw that the use of the “right” data was crucial to the learning of fine-grained ranking models.

On the algorithmic front, we first saw how concentrating on the most closely related training instances is valuable for isolating the precise visual features responsible for subtle distinctions. Our approach expanded the viability of local learning beyond traditional classification tasks to include ranking [125] (Chap. 4). Even when using only a small subset of data for training, our local model outperformed comparable global models that used all available training data. However, the isolation of each novel pair during test time also came with a higher computational cost. Next, we explored how local statistical models can address the “just noticeable difference” problem in attributes, successfully accounting for the non-uniformity of indistinguishable pairs in the feature space [127] (Chap. 5). Our local approaches outperformed the state-of-the-art (at the time of publication).

On the source data front, we approached the sparsity of supervision issue by proposing two new approaches to data augmentation using realistic synthetic examples. Specifically, we proposed a passive approach based on an automatic sampling heuristics [128] (Chap. 6), followed by an active approach based on an end-to-end adversarial network [129] (Chap. 7). We observed that sample density is distinct from sample quantity, and that even in a deep learning model, the distribution of the training data can be as important as its absolute quantity. For the active scenario, instead of having to choose from existing images as in traditional active learning, we can now directly create useful images as needed.

Finally, to accompany all the above research, we collected a brand new large-scale shoe dataset, UT-Zap50K, that is specifically designed for fine-grained comparison tasks

(Sec. 3.1 and 3.2). In the following sections, I discuss various extension ideas and future work beyond my thesis that are worthy of consideration for further research.

8.1 Extension Ideas

Here are some extension ideas that could be build upon my proposed approaches from this thesis.

- **Improved Image Generator:** Whether we form the synthetic image pairs actively or passively, the overall quality of the generated images are bounded by the quality of the image generator. It would be interesting to see how well our end-to-end model generalized to other generators, though the design of the control module would have to be modified and fine-tuned to accommodate the new generator.
- **JND with Generative Models:** While we borrow the term JND from psychophysics to motivate our research, the analogy is not 100% faithful. In particular, psychophysical experiments to elicit JND often permit systematically varying a perceptual signal until a human detects a change, e.g., slowly changing a light source until a perceptual change is registered. Unfortunately, it is infeasible to obtain such gradual (and continuous) change using the limited real images we have from existing datasets. However, such progressive spectrum *is* obtainable using the our generative framework proposed in Chapter 6. The generative capability would allow us to explore and learn the true JND in visual comparisons.
- **Dense Supervision for Classification:** In my research, I have shown the benefits of providing dense supervision through the use of synthetic images for fine-grained visual comparison. It would be interesting and a natural extension to use the same synthetic images to test whether the same can be done for fine-grained classification. A key challenge would be to determine a threshold for assigning binary attribute labels to the synthetic images. i.e., at what attribute strength does an image go from “smiling” to “not smiling”.
- **Semantic Jitter in Image Space vs. Feature Space:** Given the imperfect nature of image generators, a gap still exists between the real and the synthetic images. One potential area of exploration is to synthesize the feature layers directly instead of the images. The hypothesis here is that realistic *looking* images might

not always lead to the most useful features for learning a ranking model. This approach would bypass the need for an accurate image generator, thus reducing (or even closing) the gap.

8.2 Future Work

Here are some relevant future work that go beyond the scope of this thesis.

- **Personalized Ranking:** When working with fine-grained differences, some of the pairwise decisions could come down to personal opinions, e.g., the *comfort* level of a shoe. Currently, we take the majority vote when determining ground truth supervision from mTurk workers. It would be beneficial to have the ability to project personal bias on a pre-trained ranking model, without needing to re-train the entire model from scratch using user-specific training data. Such a personalized ranking model would especially be useful in consumer shopping applications where the ranking should ideally be customized for each user.
- **Local Deep Learning:** Local learning and deep neural networks currently do not go hand in hand. While local models aim to use only the few most relevant data samples, deep neural networks want as much data as we can provide. I explored both of these approaches in my thesis and while they might not be directly compatible, there might be ways that deep neural networks could leverage the idea of only training with the most relevant data for a given test instance as well.

Bibliography

- [1] I. Alabdulmohsin, X. Gao, and X. Zhang. Efficient active learning of halfspaces via query synthesis. In *AAAI*, 2015. 13
- [2] H. Altwaijry and S. Belongie. Relative Ranking of Facial Attractiveness. In *Winter Conference on Applications of Computer Vision (WACV)*, 2012. 1, 9
- [3] D. Angluin. Queries and concept learning. *Machine learning*, 1988. 13
- [4] C. Atkeson, A. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11:11–73, 1997. 10
- [5] S. Banerjee, A. Dubey, J. Machchhar, and S. Chakrabarti. Efficient and Accurate Local Learning for Ranking. In *ACM SIGIR Workshop*, 2009. 10
- [6] E. Baum and K. Lang. Query learning can work poorly when a human oracle is used. In *IJCNN*, 1992. 13
- [7] T. L. Berg, A. C. Berg, and J. Shih. Automatic Attribute Discovery and Characterization from Noisy Web Data. In *European Conference on Computer Vision (ECCV)*, 2010. 16
- [8] A. Biswas and D. Parikh. Simultaneous Active Learning of Classifiers and Attributes via Relative Feedback. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 1, 8, 23, 54
- [9] L. Bottou and V. Vapnik. Local learning algorithms. *Neural Computation*, 1992. 3, 10, 25, 31
- [10] C. Boutilier. *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, chapter Preference Elicitation and Preference Learning in Social Choice. Springer, 2011. 9
- [11] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2010. 2, 10
- [12] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to Rank using Gradient Descent. In *Proceedings of International Conference on Machine Learning (ICML)*, 2015. 24, 60

- [13] J. Cai, Z. Zha, M. Wang, S. Zhang, and Q. Tian. An Attribute-Assisted Reranking Model for Web Image Search. *IEEE Transactions on Image Processing*, 2015. 1
- [14] C. Cao, I. Kwak, S. Belongie, D. Kriegman, and H. Ai. Adaptive Ranking of Facial Attractiveness. In *International Conference on Multimedia and Expo (ICME)*, 2014. 1, 9
- [15] S. Chaudhuri, E. Kalogerakis, S. Giguere, , and T. Funkhouser. AttribIt: Content Creation with Semantic Attributes. *ACM Symposium on User Interface Software and Technology (UIST)*, Oct. 2013. 8
- [16] K. Chen, S. Gong, T. Xiang, and C. Loy. Cumulative Attribute Space for Age and Crowd Density Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 1, 8
- [17] L. Chen, Q. Zhang, and B. Li. Predicting Multiple Attributes via Relative Multi-task Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 8
- [18] A. Datta, R. Feris, and D. Vaquero. Hierarchical Ranking of Facial Attributes. In *Face and Gesture*, 2011. 1, 8, 9
- [19] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. Dhillon. Information-Theoretic Metric Learning. In *International Conference on Machine Learning (ICML)*, 2007. 28, 30
- [20] B. Demirel, R. G. Cinbis, and N. Ikingler-Cinbis. Attributes2Classname: A Discriminative Model for Attribute-Based Unsupervised Zero-Shot Learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 8
- [21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 54
- [22] M. Dixit, R. Kwitt, M. Niethammer, and N. Vasconcelos. AGA: Attribute-Guided Augmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 11, 12
- [23] C. Domeniconi and D. Gunopulos. Adaptive Nearest Neighbor Classification using Support Vector Machines. In *Conference on Neural Information Processing Systems (NIPS)*, 2001. 10
- [24] A. Dosovitskiy, J. Springenberg, and T. Brox. Learning to generate chairs with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 11

- [25] A. Dosovitskiy, J. Springenberg, M. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 6, 62, 75
- [26] K. Duh and K. Kirchhoff. Learning to Rank with Partially-Labeled Data. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2008. 10
- [27] Q. Fan, P. Gabbur, and S. Pankanti. Relative Attributes for Large-Scale Abandoned Object Detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013. 1, 9
- [28] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing Objects by their Attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 1
- [29] R. Farrell, O. Oza, N. Zhang, V. Morariu, T. Darrell, and L. Davis. Birdlets: Subordinate Categorization using Volumetric Primitives and Pose-Normalized Appearance. In *International Conference on Computer Vision (ICCV)*, 2011. 2, 10
- [30] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(9), September 2010. 13
- [31] D. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2002. 39, 40
- [32] A. Freytag, E. Rodner, and J. Denzler. Selecting influential examples: active learning with expected model output changes. In *ECCV*, 2014. 6, 13
- [33] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning Globally-Consistent Local Distance Functions for Shape-Based Image Retrieval and Classification. In *International Conference on Computer Vision (ICCV)*, 2007. 29
- [34] X. Geng, T. Liu, T. Qin, A. Arnold, H. Li, and H. Shum. Query Dependent Ranking using K-nearest Neighbor. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2008. 10
- [35] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 11
- [36] K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. DRAW: A Recurrent Neural Network for Image Generation. In *Proceedings of International Conference on Machine Learning (ICML)*, 2015. 11

- [37] B. Hariharan and R. Girshick. Low-Shot Visual Recognition by Shrinking and Hallucinating Features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 11, 12
- [38] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *PAMI*, 18:607–616, 1996. 10
- [39] S. Hauberg, O. Freifeld, A. Larsen, J. Fisher, and L. Hansen. Dreaming more data: Class-independent distributions over diffeomorphisms for learned data augmentation. In *AISTATS*, 2016. 12
- [40] S. Hauberg, O. Freifeld, A. Larsen, J. Fisher, and L. Hansen. Dreaming More Data: Class-dependent Distributions over Diffeomorphisms for Learned Data Augmentation. In *International Conference on Artificial Intelligence and Statistics*, 2017. 11
- [41] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 75
- [42] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. Technical report, University of Massachusetts, Amherst, 2007. 19
- [43] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 11
- [44] M. W. Huijser and J. C. van Gemert. Active decision boundary annotation with deep generative models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 13
- [45] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 11
- [46] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. In *NIPS 14 Deep Learning workshop*, 2014. 12
- [47] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 60
- [48] P. Jain, B. Kulis, and K. Grauman. Fast Image Search for Learned Metrics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 29

- [49] T. Joachims. Optimizing Search Engines using Clickthrough Data. In *Knowledge Discovery in Databases (PKDD)*, 2002. 8, 23
- [50] D. B. Judd. Chromaticity Sensibility to Stimulus Differences. *Journal of the Optical Society of America*, 22(2):72–72, Feb 1932. 4
- [51] M. M. Kalayeh, B. Gong, and M. Shah. Improving Facial Attribute Prediction using Semantic Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1
- [52] A. Kapoor, P. Jain, and R. Viswanathan. Multilabel Classification using Bayesian Compressed Sensing. In *Conference on Neural Information Processing Systems (NIPS)*, 2012. 37
- [53] I. Kemelmacher-Shlizerman, S. Suwajanakorn, and S. Seitz. Illumination-aware age progression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 11
- [54] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele. Lucid data dreaming for object tracking. Technical Report arXiv:1703.09554, 2017. 12
- [55] A. Khosla, W. A. Bainbridge, A. Torralba, and A. Oliva. Modifying the memorability of face photographs. In *International Conference on Computer Vision (ICCV)*, 2013. 11
- [56] D. Kingma and M. Welling. Auto-encoding variational bayes. In *Proceedings Int. Conf. on Learning Representations (ICLR)*, 2014. 11
- [57] A. Kovashka and K. Grauman. Attribute Adaptation for Personalized Image Search. In *International Conference on Computer Vision (ICCV)*, 2013. 9
- [58] A. Kovashka and K. Grauman. Attribute Pivots for Guiding Relevance Feedback in Image Search. In *International Conference on Computer Vision (ICCV)*, 2013. 1, 44, 45
- [59] A. Kovashka, D. Parikh, and K. Grauman. WhittleSearch: Image Search with Relative Attribute Feedback. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1, 8, 9, 16, 20, 21, 33, 51, 52
- [60] A. Kovashka, D. Parikh, and K. Grauman. WhittleSearch: Interactive Image Search with Relative Attribute Feedback. *International Journal of Computer Vision (IJCV)*, 115(2):185–210, Nov 2015. 8, 17, 23, 54

- [61] T. Kulkarni, W. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 11
- [62] N. Kumar, P. Belhumeur, and S. Nayar. FaceTracer: A Search Engine for Large Collections of Images with Faces. In *European Conference on Computer Vision (ECCV)*, 2008. 1
- [63] N. Kumar, A. Berg, P. Belhumeur, and S. Nayar. Attribute and Simile Classifiers for Face Verification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009. 20, 21, 62, 63
- [64] R. Kwitt, S. Hegenbart, and M. Niethammer. One-Shot Learning of Scene Locations via Feature Trajectory Transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 11, 12
- [65] P. Laffont, Z. Ren, X. Tao, C. Qian, and J. Hays. Transient attributes for high-level understanding and editing of outdoor scenes. In *SIGGRAPH*, 2014. 11
- [66] C. Lampert, H. Nickisch, and S. Harmeling. Learning to Detect Unseen Object Classes by Between-Class Attribute Transfer. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 1
- [67] G. Lample, N. Zeghidour, N. Usunier, A. Bordes, L. DENOYER, et al. Fader networks: Manipulating images by sliding attributes. In *NIPS*, pages 5963–5972, 2017. 11
- [68] M. Li, W. Zuo, and D. Zhang. Convolutional network for attribute-driven and identity-preserving human face generation. Technical Report arXiv:1608.06434, 2016. 11
- [69] S. Li, S. Shan, and X. Chen. Relative Forest for Attribute Prediction. In *Asian Conference on Computer Vision (ACCV)*, 2012. 1, 8, 9, 20, 21, 30, 31, 33, 44
- [70] L. Liang and K. Grauman. Beyond comparing image pairs: Setwise active learning for relative attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 13
- [71] H. Lin, C. Yu, and H. Chen. Query-Dependent Rank Aggregation with Local Models. In *Asia Information Retrieval Societies Conference (AIRS)*, 2011. 10
- [72] L. Maaten and G. Hinton. Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research (JMLR)*, 9:2579–2605, 2008. 48, 82
- [73] S. Maji. Discovering a lexicon of parts and attributes. In *Second International Workshop on Parts and Attributes, ECCV*, 2012. 18

- [74] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-Grained Visual Classification of Aircraft. Technical Report arXiv:1306.5151, 2013. 2, 10
- [75] T. Matthews, M. Nixon, and M. Niranjan. Enriching texture analysis with semantic data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 1, 9
- [76] Z. Meng, N. Adluru, H. J. Kim, G. Fung, and V. Singh. Attribute-guided face generation using conditional cycleGAN. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2018. 11
- [77] Z. Meng, N. Adluru, H. J. Kim, G. Fung, and V. Singh. Efficient Relative Attribute Learning using Graph Neural Networks. In *Proceedings of European Conference on Computer Vision (ECCV)*, September 2018. 8
- [78] E. Miller, N. Matsakis, and P. Viola. Learning from One Example through Shared Densities on Transforms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000. 11
- [79] S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 11
- [80] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 11
- [81] P. O’Donovan, J. Libeks, A. Agarwala, and A. Hertzmann. Exploratory Font Selection Using Crowdsourced Attributes. In *SIGGRAPH*, 2014. 8
- [82] A. Oliva and A. Torralba. Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *International Journal of Computer Vision (IJCV)*, 42:145–175, 2001. 21
- [83] G. Pandey and A. Dukkipati. Variational methods for conditional multimodal learning: Generating human faces from attributes. Technical Report arXiv:1603.01801, 2016. 11
- [84] D. Parikh and K. Grauman. Relative Attributes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011. 1, 8, 9, 20, 21, 23, 24, 30, 31, 33, 39, 44, 54, 62, 65, 66, 79
- [85] D. Park and D. Ramanan. Articulated pose estimation with tiny synthetic videos. In *ChaLearn Workshop, CVPR*, 2015. 12

- [86] E. Parzen. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962. 43
- [87] M. Paulin, J. Revaud, Z. Harchaoui, F. Perronnin, and C. Schmid. Transformation pursuit for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 12
- [88] X. Peng, B. Sun, K. Ali, and K. Saenko. Learning deep object detectors from 3d models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 12
- [89] X. Peng, Z. Tang, F. Yang, R. S. Feris, and D. Metaxas. Jointly optimize data augmentation and network training: Adversarial data augmentation in human pose estimation. In *CVPR*, 2018. 12
- [90] L. Pishchulin, A. Jain, C. Wojek, T. Thormahlen, and B. Schiele. In good shape: Robust people detection based on appearance and shape. In *British Machine Vision Conference (BMVC)*, 2011. 12
- [91] B. Qian, X. Wang, F. Wang, H. Li, J. Ye, and I. Davidson. Active learning from relative queries. In *IJCAI International Joint Conference on Artificial Intelligence*, 2013. 13
- [92] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 11
- [93] D. Reid and M. Nixon. Human identification using facial comparative descriptions. In *ICB*, 2013. 8
- [94] D. Reid and M. Nixon. Using Comparative Human Descriptions for Soft Biometrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 36, 2014. 1, 8
- [95] A. Sadvnik, A. Gallagher, D. Parikh, and T. Chen. Spoken Attributes: Mixing Binary and Relative Attributes to Say the Right Thing. In *International Conference on Computer Vision (ICCV)*, 2013. 1, 44
- [96] R. Sandeep, Y. Verma, and C. Jawahar. Relative Parts: Distinctive Parts for Learning Relative Attributes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1, 8, 9, 19, 20, 44, 46
- [97] W. Scheirer, N. Kumar, P. Belhumeur, and T. Boult. Multi-Attribute Spaces: Calibration for Attribute Fusion and Similarity Search. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 9

- [98] E. Schwartz, L. Karlinsky, J. Shtok, S. Harary, M. Marder, R. Feris, A. Kumar, R. Giryes, and A. M. Bronstein. Delta-encoder: An effective sample synthesis method for few-shot object recognition. Technical Report arXiv:1806.04734, 2018. 12
- [99] B. Settles. Active learning literature survey. Technical report, 2010. 13
- [100] G. Shakhnarovich, P. Viola, and T. Darrell. Fast Pose Estimation with Parameter-Sensitive Hashing. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2003. 12
- [101] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 12
- [102] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016. 13
- [103] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from Simulated and Unsupervised Images through Adversarial Training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 12
- [104] A. Shrivastava, S. Singh, and A. Gupta. Constrained semi-supervised learning using attributes and comparative attributes. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2012. 1, 8
- [105] B. Siddiquie, R. Feris, and L. Davis. Image Ranking and Retrieval based on Multi-Attribute Queries. In *CVPR*, 2011. 1
- [106] P. Simard, D. Steinkraus, and J. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *ICDAR*, 2003. 6
- [107] K. Singh and Y. J. Lee. End-to-end localization and ranking for relative attributes. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016. 6, 8, 9, 23, 24, 60, 61, 62, 64, 65, 66, 70, 75, 76, 77, 78, 79
- [108] Y. Souri, E. Noury, and E. Adeli. Deep relative attributes. In *Asian Conference on Computer Vision (ACCV)*, 2016. 6, 8, 23, 24, 66
- [109] J.-C. Su, C. Wu, H. Jiang, and S. Maji. Reasoning about Fine-Grained Attribute Phrases using Reference Games. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 8

- [110] A. Torralba. Contextual Priming for Object Detection. *International Journal of Computer Vision (IJCV)*, 53(2):169–191, 2003. 23, 29, 62
- [111] P. Upchurch, J. G. G. Pleiss, R. Pless, N. Snavely, K. Bala, and K. Weinberger. Deep feature interpolation for image content changes. In *CVPR*, 2017. 11
- [112] G. Varol, J. Romero, X. Martin, N. Mahmood, M. J. Black, I. Laptev, and C. Schmid. Learning from synthetic humans. In *CVPR*, 2017. 12
- [113] S. Vijayanarasimhan and K. Grauman. What’s it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 6, 13
- [114] S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. *International Journal of Computer Vision (IJCV)*, 108(1):97–114, May 2014. 6, 13
- [115] P. Vincent and Y. Bengio. K-Local Hyperplane and Convex Distance Nearest Neighbor Algorithms. In *Conference on Neural Information Processing Systems (NIPS)*, 2001. 10
- [116] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of International Conference on Machine Learning (ICML)*, 2008. 6
- [117] F. Xiao and Y. J. Lee. Discovering the spatial extent of relative attributes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 9, 66
- [118] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2Image: Conditional Image Generation from Visual Attributes. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016. 11, 56, 57, 62, 63, 65, 70, 72, 75
- [119] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2Image: Conditional Image Generation from Visual Attributes. Technical Report arXiv:1512.00570, 2016. 11, 56, 57, 70, 72
- [120] D. Yang and J. Deng. Shape from shading through shape evolution. Technical Report arXiv:1712.02961, 2017. 12
- [121] L. Yang, P. Luo, C. C. Loy, and X. Tang. A Large-Scale Car Dataset for Fine-Grained Categorization and Verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2, 10
- [122] X. Yang, T. Zhang, C. Xu, S. Yan, M. Hossain, and A. Ghoneim. Deep relative attributes. *IEEE Trans. on Multimedia*, 18(9), Sept 2016. 6, 8, 23, 24, 58

- [123] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei. Boosting Image Captioning with Attributes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 1
- [124] A. Yu and K. Grauman. 3, 5, 22, 41
- [125] A. Yu and K. Grauman. Fine-Grained Visual Comparisons with Local Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 3, 6, 22, 62, 64, 65, 66, 79, 87
- [126] A. Yu and K. Grauman. Predicting Useful Neighborhoods for Lazy Local Learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 3, 6, 22, 35, 36, 38
- [127] A. Yu and K. Grauman. Just Noticeable Differences in Visual Attributes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 5, 6, 41, 87
- [128] A. Yu and K. Grauman. Semantic Jitter: Dense Supervision for Visual Comparisons via Synthetic Images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 6, 55, 79, 87
- [129] A. Yu and K. Grauman. Thinking Outside the Pool: Active Training Image Creation for Relative Attributes. Technical Report arXiv:1901.02551, 2019. 6, 68, 87
- [130] M. E. Yumer, S. Chaudhuri, J. K. Hodgins, and L. B. Kara. Semantic Shape Editing Using Deformation Handles. *ACM Transactions on Graphics*, 34(4):86:1–86:12, July 2015. 8
- [131] G. Zhang, M. Kan, S. Shan, and X. Chen. Generative adversarial network with spatial attention for face attribute editing. In *ECCV*, 2018. 11
- [132] H. Zhang, A. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. 10, 31
- [133] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017. 11
- [134] Y. Zhang, S. Song, E. Yumer, M. Savva, J. Lee, H. Jin, and T. Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. In *CVPR*, 2017. 12
- [135] L. Zhao, G. Sukthankar, and R. Sukthankar. Robust active learning using crowdsourced annotations for activity recognition. In *HCOMP*, 2011. 6, 13

- [136] J.-J. Zhu and J. Bento. Generative adversarial active learning. Technical Report arXiv:1702.07956, 2017. 13
- [137] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 11