Describable Visual Attributes for Face Images

Neeraj Kumar

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2011

ABSTRACT

Describable Visual Attributes for Face Images

Neeraj Kumar

We introduce the use of *describable visual attributes* for face images. Describable visual attributes are labels that can be given to an image to describe its appearance. This thesis focuses mostly on images of faces and the attributes used to describe them, although the concepts also apply to other domains. Examples of face attributes include gender, age, jaw shape, nose size, *etc.* The advantages of an attribute-based representation for vision tasks are manifold: they can be composed to create descriptions at various levels of specificity; they are generalizable, as they can be learned once and then applied to recognize new objects or categories without any further training; and they are efficient, possibly requiring exponentially fewer attributes (and training data) than explicitly naming each category. We show how one can create and label large datasets of real-world images to train classifiers which measure the presence, absence, or degree to which an attribute is expressed in images. These classifiers can then automatically label new images.

We demonstrate the current effectiveness and explore the future potential of using attributes for image search, automatic face replacement in images, and face verification, via both human and computational experiments. To aid other researchers in studying these problems, we introduce two new large face datasets, named FaceTracer and PubFig, with labeled attributes and identities, respectively.

Finally, we also show the effectiveness of visual attributes in a completely different domain: plant species identification. To this end, we have developed and publicly released the *Leafsnap* system, which has been downloaded by almost half a million users. The mobile phone application is a flexible electronic field guide with high-quality images of the tree species in the Northeast US. It also gives user instant access to our automatic recognition system, greatly simplifying the identification process.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I have been incredibly fortunate to have been co-advised by both Peter Belhumeur and Shree Nayar. Not only are they brilliant researchers, but also incredibly warm people with only my best interests at heart. Through the course of my PhD, they have helped me evolve from a clueless student to a capable researcher. Their insight into fundamental research problems and skill at coming up with startlingly original solutions continues to amaze me, even after six years of working with them. Perhaps most important (because of its relative dearth in our field) has been their emphasis on clear and effective communication: crafting beautiful prose for research papers and grant applications alike, carefully planning talks for maximum simplicity, and honing both skills to perfection with hard work and lots of practice.

Having even one advisor with this kind of skill and dedication would be incredible enough for any grad student, but I've had two! Peter, as my more-than-full-time advisor for the past few years, I thank you most especially for looking at the world so differently from me – it has really opened my eyes and changed the way I look at it as well. Shree, as my primary advisor for the first few years (and recently as advisor-at-large), I thank you for knowing me better than I know myself and helping me navigate the emotional aspects of a PhD. It has been a real pleasure and honor to work with both of you, and I sincerely thank you for these wonderful years together.

Second, I'd like to thank my colleagues and friends in the department, who I've spent so much time talking to, learning from, and bonding with: Jinwei Gu, for putting up with my endless distractions, and for your inspirational work ethic; Thomas Berg, for our many discussions about programming in general and our menagerie of code in particular, and also for shouldering the burden of all the boring tasks I shirked; Mohit Gupta, for patiently explaining what postdoc life is *really* like and preparing me for the life ahead; Alex Berg and Li Zhang, for being encouraging mentors and giving me someone to look up to; Ollie, Changyin, Sujit, Kshitiz, and all the other CAVE lab and Vaplab members over the years who have engaged me in countless thought-provoking conversations about science and research and life; and finally, Anne Fleming, for being unfailingly helpful for every administrative need I had, and for excusing my often inexcusable delays in filing important paperwork.

Outside my immediate academic group, I want to thank the many other researchers who have helped me mature as a scientist: David Jacobs, for being at times my unofficial third advisor, showing me that people can have radically different research styles and be just as successful; David Kriegman, Serge Belongie, and Florian Schroff, for working long-distance with me on a number of exciting projects where I could spend more time thinking and less time coding; Terry and Walter, for dealing with my perpetual tardiness when I had my AFS hat on; and finally Tony Jebara and Shih-Fu Chang, for being my committee members (Tony since the very beginning, with my candidacy and proposal as well).

Of course, life is not all work, and that is thanks largely to my friends and family in New York, who've kept me (relatively) sane all these years: Rob and Sheila, for being my life support here, and Rob especially for being a fellow hacker; Niketa, for reminding me what a wonderful sister I have and how much I'm going to miss her once I leave New York; Mike, Kapil, Tim, and Vishakh, for being my concert buddies, and Mike & Kapil especially for our many fascinating lunches together; Anoop Bhaiya and family, for being my home-away-from-home; and of course Steph, Meenal, Vivek, Baktash, Sonal, Alka and everyone else who's made my life better here, enduring my endless ramblings on music, film, literature, and other things I have no authority on.

Finally, as the saying goes, I would not be here were it not for my parents. Well, of course that's true, but really: I can't imagine having reached this point in my life were it not for their endless love and support over the years. They deserve all the credit for my success. Above all else, though, I thank them for showing me that doing what you love is the best way to live. I could not agree more.

This thesis is dedicated to my parents, for everything.

Chapter 1

Introduction

One of history's most successful books was a five-volume pharmacopoeia titled *De Materia Medica*, written in the first century by the Greek botanist and physician Pedanius Dioscorides. It is perhaps the earliest known field guide, with pictures and written descriptions of nearly 600 plant species, showing how each could be found and identified, along with their uses for medicinal purposes. This work would be the first in a line of botanical texts, including the ninth century medieval agricultural and toxicological texts of Ibn Washiyah, and the early eighteenth century *Systema Naturae* of Carl Linneaus, which laid out the rules of modern taxonomy.

All of these works have in common an effort to teach the reader how to identify a plant (or animal) by describable aspects of its visual appearance. Consider the description in the caption of Fig. 1.1 where thirty words serve to carefully describe the maple tree, its bark, and its leaf. This identification explicitly uses an intermediate representation of describable visual attributes, *e.g.*, "5-lobed," "as wide as long," "sharp teeth," "bark medium-gray," *etc.* It is possible to recognize such visual attributes independently and to use many of them in combination to describe and identify species.

While the use of describable visual attributes for identification has been around since antiquity, it has not been the focus of work by researchers in computer vision and related disciplines. Most existing methods for recognition (*e.g.*, [Sivic and Zisserman, 2003; Nister and Stewenius, 2006; Nowak *et al.*, 2006; Dalal and Triggs, 2005]) work by extracting low-level features in images, such as pixel values, gradient directions, histograms of oriented

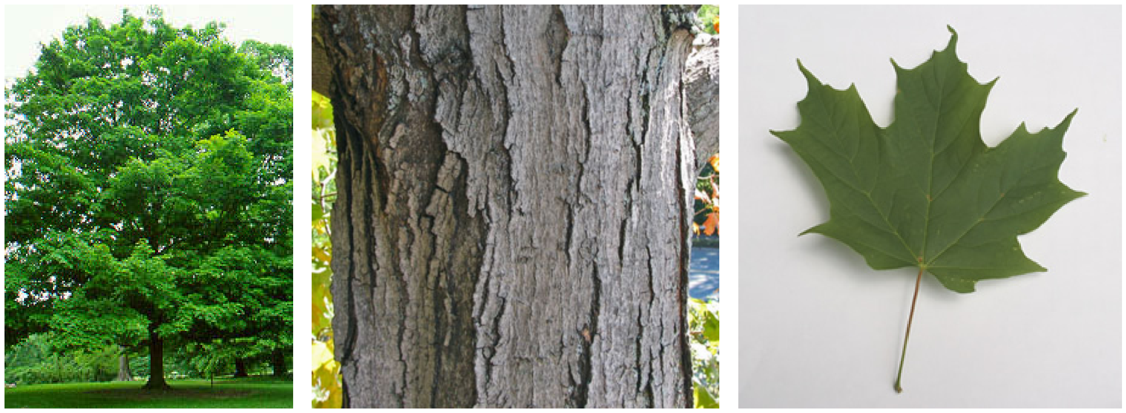


Figure 1.1: Images of a maple tree, its bark, and a single maple leaf. Note how accurately they are described by this 30-word description from the *Manual of Vascular Plants of Northeastern United States*: "...tree to 40 m; bark medium-gray, becoming roughened with loose-edged plates; leaves flat, about as wide as long,...5-lobed with rounded sinuses, the lobes usually bearing a few large sharp teeth..." We use the concept of such describable attributes for performing search, replacement, and identification of faces in images.

gradients [Dalal and Triggs, 2005], SIFT [Lowe, 2003], *etc.*, which are then used to *directly* train classifiers for identification or detection. Similarly for search and retrieval applications, such low-level descriptors are often used directly to find similar images [Sivic and Zisserman, 2003; Nister and Stewenius, 2006].

In contrast, we use low-level image features to first learn *intermediate representations*, in which images are labeled with an extensive list of descriptive visual attributes. For faces, attributes can range from simple demographic information such as gender, age, or ethnicity; to physical characteristics of a face such as nose size, mouth shape, or eyebrow thickness; and even to environmental aspects such as lighting conditions, facial expression, or image quality.

Although these attributes could clearly be useful in a variety of domains (such as object recognition, species identification, architectural description, action recognition, *etc.*), we focus on faces as they are perhaps the most common objects in images. Photos of faces are everywhere: cell phones, personal photo collections, social networking sites, news feeds, movies, license and passport databases, surveillance videos – the list goes on and on. Our own work has shown that there are on average 0.4 faces per image in the billions of images posted to the Internet [Kumar *et al.*, 2008]. The recent success of methods for reliably detecting faces in images [Viola and Jones, 2001; Huang *et al.*, 2007a] has increased the promise of face-related applications, and in this work we show that it is possible to automatically and reliably estimate attributes for face images.

Why might one need an intermediate layer of attributes? What do they afford? Why not train classifiers directly for the task at hand? Visual attributes – much like words – are composable, offering tremendous flexibility and efficiency. Attributes can be combined to produce descriptions at multiple levels, including object categories, objects, or even instances of objects. For example, one can describe "white male" at the category level (a set of people), or "white male brown-hair green-eyes scar-on-forehead" at the object level (a specific person), or add "..., smiling lit-from-above seen-from-left" to the previous for an instance of the object (a particular image of a person).

Moreover, attributes are generalizable; one can learn a set of attributes from large image collections and then apply them in almost arbitrary combinations to novel images, objects, or categories. This makes them an attractive solution to the "zero-shot learning" problem, where one has to recognize an object or category that one has never seen before.

Better still, attributes are efficient: consider that k binary attributes may suffice to identify 2^k categories, clearly more efficient than naming each category individually. (Of course, in practice, the potential benefits are limited by the problem domain, the type of categories being considered, and the accuracy of learned classifiers.) In contrast to existing labeling efforts such as ImageNet [Deng *et al.*, 2009] and LabelMe [Russell *et al.*, 2008] that label large collections of images by category or object name, the use of attributes may provide a significantly more compact way of describing objects. This would allow for the use of much smaller labeled datasets to achieve comparable performance on recognition tasks.

Another advantage of attributes is that they can be used to relate objects in non-hierarchical and possibly quite complex ways, unlike traditional mutually-exclusive hierarchies of objects, such as Caltech 256 [Griffin *et al.*, 2007] and ImageNet [Deng *et al.*, 2009]. This has important conseqeuences for most learning-based approaches to vision that require several trainig instances per category to be learned. For many categories at the leaves of traditional hierarchies, such as a particular individual or a specific species of mushroom, it might be infeasible to obtain a large amount of training data. This problem is often exacerbated by the fact that for fine-scale categories with high intra-class variation, such as different types of maples, the number of examples to reliably distinguish between classes might be prohibitively high.

However, by describing the object using a collection of attributes, we can sidestep this issue by using many disparate objects or categories, *all of which share a particular attribute*, as training data. In addition, particularly for natural phenomena such as plants and animals, different fine-scale categories might be most readily distinguished precisely on the basis of a few attributes – making them particularly suitable for this task.

Perhaps most importantly, these attributes can be chosen to align with the domain-appropriate vocabulary that people have developed over time for describing different types of objects. For faces, this includes descriptions at the coarsest level (such as gender and age) to more subtle aspects (such as expressions and shape of face parts) to highly face-specific marks (such as moles and scars). One could use these attributes to search through all the images in a large photo collection. For instance, a father might search his private photo collection by typing "smiling blond girl outside with frowning older boy" to retrieve all vacation photos of their two children. Or a law enforcement officer might use a witness' description of a suspect – "white male, thirties, scar-on-forehead, blue-eyes." to retrieve matching photos from a database of felons. In the domain of plant species identification, a science teacher in a park might photograph a leaf using a mobile phone application, then type "with yellow-bark" to retrieve an exact match for the English Plane tree along with species distribution in the world and a species map for all other such trees in the park.

In this thesis, we describe how to harness the power of attributes in the domain of faces. In our approach, diagrammed in Fig. 1.2, an extensive vocabulary of visual attributes is used to label a large dataset of images, which is then used to train classifiers that *automatically* recognize the presence, absence, or degree to which these attributes are exhibited in new images. The classifier outputs can then be used to search through large image collections, perform automatic face replacement, identify faces, and they also seem promising for use in many other tasks such as image exploration or automatic description-generation.

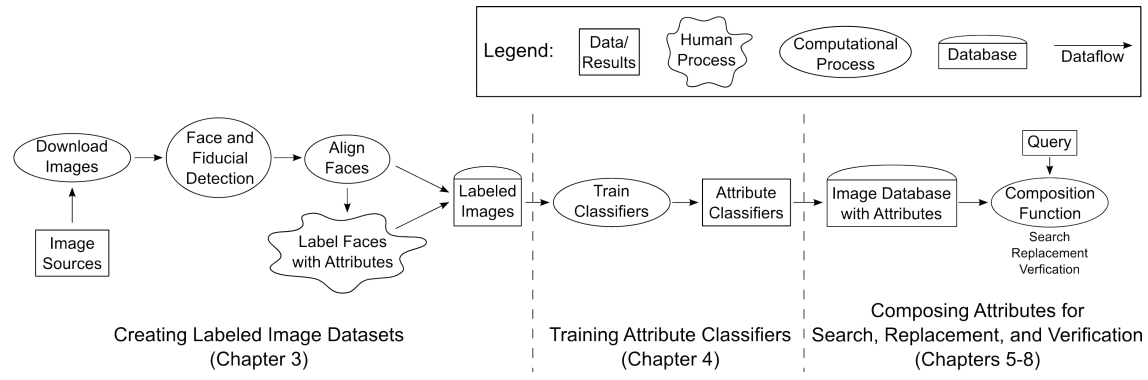

Legend:
Data/ Results
Human Process
Computational Process
Database
Dataflow
Download Images
Image Sources
Face and Fiducial Detection
Align Faces
Label Faces with Attributes
Labeled Images
Train Classifiers
Attribute Classifiers
Image Database with Attributes
Query
Composition Function
Search
Replacement
Verfication
Creating Labeled Image Datasets (Chapter 3)
Training Attribute Classifiers (Chapter 4)
Composing Attributes for Search, Replacement, and Verification (Chapters 5-8)

Figure 1.2: A diagram of our system architecture. We download images from various image sources on the internet and label them with attributes to create labeled image datasets. We use these to train attribute classifiers that can then be used for a variety of applications including search, replacement, and verification by defining appropriate composition functions.

1.1 Creating Labeled Image Datasets (Chapter 3)

Two recent trends in internet services have made collecting and labeling image data dramatically easier. The first, large internet photo-sharing sites such as flickr.com and picasa.com are growing exponentially and host billions of public images, many with textual annotations and comments. The second, efficient marketplaces for small amounts of labor such as Amazon's Mechanical Turk (MTurk) [Amazon, 2011] make it possible to purchase small amounts of web-based labeling effort at very low overhead.

We envision a new thrust for image collection and labeling of describable visual attributes of objects. While several existing efforts in the computer vision community have exploited services for collecting and labeling images (*e.g.*, ImageNet [Deng *et al.*, 2009] and LabelMe [Russell *et al.*, 2008]), their focus has been on naming objects, images, and regions of images using nouns – which is quite different from our visual attributes. One difference is that attributes need not be binary – a person's height or the transparency of a leaf are both continuous attributes. Another critical difference is that visual attributes can be composed more freely than names which generally exist in a tree structured hierarchy. A set of general attributes can be combined in an exponential number of ways to describe many objects at different levels of specificity. Attributes can therefore compactly provide a great deal of information, both about object properties and their identity.

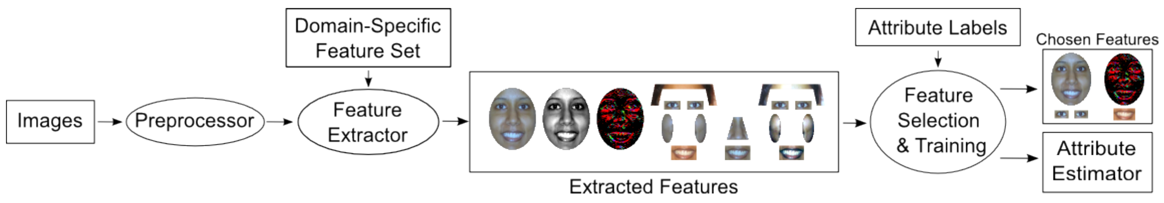


Figure 1.3: Attribute training architecture. Images are preprocessed using a domain-specific preprocessor (for faces this includes face detection and affine alignment). Various low-level features are then extracted from labeled images and use for an automatic feature selection process which finds the best features to use for a given attribute. The output is an attribute estimator and an associated set of low-level features used for this attribute.

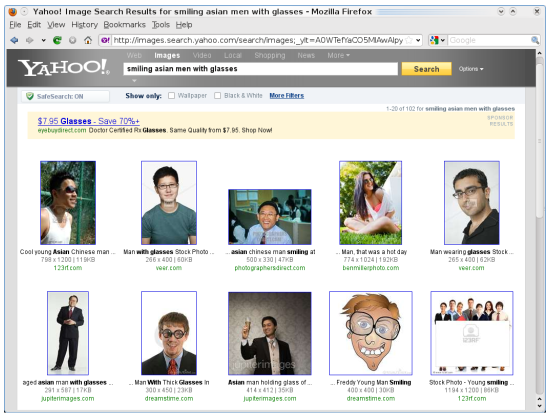
1.2 Training Attribute Classifiers (Chapter 4)

We use feature selection to find an appropriate set of features for use in constructing the attribute classifier. A schematic of our architecture is shown in Fig. 1.3. The key idea is to leverage the many efficient and effective low-level features that have been developed by the computer vision community, choosing amongst a large set of them to find the ones suited for learning a particular attribute. Examples of low-level features include histograms of orientation gradients, which have proven to be very useful in a number of leading vision techniques including SIFT [Lowe, 2003]; edge magnitudes and curvature measurements, many of which form the backbone for various shape-based methods [Belongie *et al.*, 2002; Ling and Jacobs, 2007]; and color moments, key components in many image retrieval algorithms [Flickner *et al.*, 1995; Cox *et al.*, 2000].

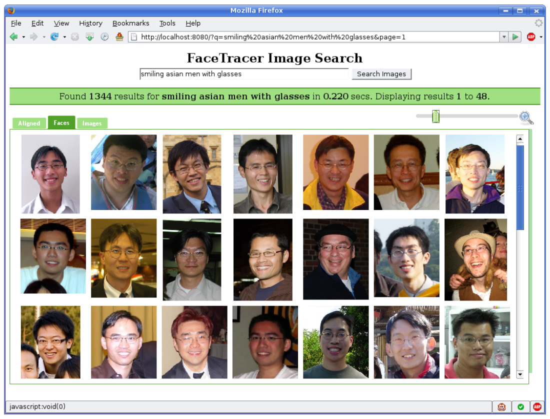
Formally, the learning process for attribute classifier $\mathbf{a_i}$ can be described as follows. A detected face image X is first aligned to create input representation X'. Then, a set of k low-level feature vectors $\mathbf{f_j}$ are extracted from the aligned image to form a feature set $\mathcal{F}(X)$:

$$\mathcal{F}(X) = \left\{\mathbf{f_1}(X'), \cdots, \mathbf{f_k}(X')\right\}. \tag{1.1}$$

For the given attribute, a feature selection algorithm chooses an appropriate set of features $\mathcal{F}_i$ from this set. Due to the large number of possible features and attributes, it is necessary to perform feature selection automatically and efficiently. We select features in an iterative manner, such that an attribute classifier $\mathbf{a_i}$ learned using the training set and the current


Yahoo! Image Search Results for smiling asian men with glasses - Mozilla Firefox
smiling asian men with glasses
Search

(a) Yahoo image search results


FaceTracer Image Search
smiling asian men with glasses
Search Images
Found 1344 results for smiling asian men with glasses in 0.220 secs. Displaying results 1 to 48.

(b) Attribute-based image search results

Figure 1.4: Results for the query, "smiling asian men with glasses," using (a) the Yahoo image search engine (as of November 2010) and (b) our face search engine. Conventional image search engines rely on text annotations, such as file metadata, manual labels, or surrounding text, which are often incorrect, ambiguous, or missing. In contrast, we use attribute classifiers to *automatically* label images with faces in them, and store these labels in a database. At search time, only this database needs to be queried, and results are returned instantaneously. The attribute-based search results are much more relevant to the query.

set of features has least error (subject to regularization to prevent over-fitting). The output of this process is both the set of chosen features $\mathcal{F}_i$ and the learned attribute classifier $\mathbf{a_i}$. We note that this procedure is not optimal; picking optimal features in a non-linear setting is still an open problem in machine learning.

Measuring the attributes on a new image is then a matter of performing alignment, extracting features $\mathcal{F}_i$, and evaluating the attribute classifier. This process is highly parallelizable, both by data (individual images) and by task (preprocessing, feature extraction, classifier evaluation), leading to fast implementations that lend themselves to large-scale and interactive applications, described next.

1.3 Face Search (Chapter 5)

The first application of describable visual attributes is image search. Currently deployed image search engines such as Google Images and Flickr are mainly limited to searching for images based only on surrounding text and metadata such as links, file names, etc. This is a significant limitation – the vast majority of images available online and in large photo collections are not annotated in any meaningful way, rendering them effectively invisible to these search engines. In contrast, attributes allow for searching through images even without metadata, albeit only in domains for which we have trained attribute classifiers. Attributes have the added advantage of being tuned for particular domains depending on the vocabularies used. For example, botanists could use a plant search engine to find plant species based on textual descriptions, or use the descriptions to augment example-based queries. Or, home-buyers could use an image search engine to find houses built in a certain style, or containing specified architectural features.

Looking concretely at the problem of face search, Fig. 1.4a show the results of the query, "smiling asian men with glasses," using a conventional image search engine (Yahoo Image Search, as of November 2010). Yahoo's reliance on text annotations causes it to find some images that have no relevance to the query. In addition, many of the correct results on Yahoo point to stock photography websites, which can afford to manually label their images with keywords – but only because they have collections of a limited size, and they label only the coarsest attributes. Clearly, this approach does not scale to the entire internet.

In contrast, Fig. 1.4b shows the results for the same query using a search engine built using our attribute outputs. These are computed by simply taking the product of the given attribute values for all images in a database and returning the top-ranked images. The difference in quality of search results is clearly visible – our system returns only the images that match the query. Note that one could just as easily apply this to arbitrary images on the web as to those shared in a social network such as Facebook or even a user's personal photo collection.

An exciting aspect of describable visual attributes is their direct meaning to users, unlike many low-level features. This allows users to provide more effective feedback for improving search ranking. Not only can a user specify re-ranking of results, but they can explicitly tell us what attributes were important for their decision. This dovetails well with recent work in machine learning on taking user preference into account when learning rankings (*e.g.*, [Jin *et al.*, 2008; Zheng *et al.*, 2008]). Also, recent work has begun to address ranking for structured data [Qin *et al.*, 2008].

1.4 Face Replacement (Chapter 6)

While the size and availability of image collections online is leading to many exciting new applications, such as the one just described, it is also creating new problems. One of the most important of these is privacy. Online systems such as Google Street View [1] and EveryScape [2] allow users to interactively navigate through panoramic images of public places created using thousands of photographs. Many of the images contain people who have not consented to be photographed, much less to have these photographs publicly viewable. Identity protection by obfuscating the face regions in the acquired photographs using blurring, pixelation, or simply covering them with black pixels is often undesirable as it diminishes the visual appeal of the image. Furthermore, many of these methods are currently applied manually, on an image-by-image basis. Since the number of images being captured is growing rapidly, not to mention the explosion of videos being put on sites such as youtube or justin.tv, any manual approach will soon be intractable. We believe that an attractive solution to the privacy problem is to remove the identities of people in photographs by automatically replacing their faces with ones from a collection of stock images.

Automatic face replacement has other compelling applications as well. For example, people commonly have large personal collections of photos on their computers. These collections often contain many photos of the same person(s) taken with different expressions, and under various poses and lighting conditions. One can use such collections to create novel images by replacing faces in one image with more appealing faces of the same person from other images. Finally, when taking group shots, the "burst" mode available in most cameras can be used to capture several images at once. With an automatic face replacement approach that was attribute-aware, one could then create a single composite image with,

[1] `http://maps.google.com/help/maps/streetview/`

[2] `http://www.everyscape.com/`

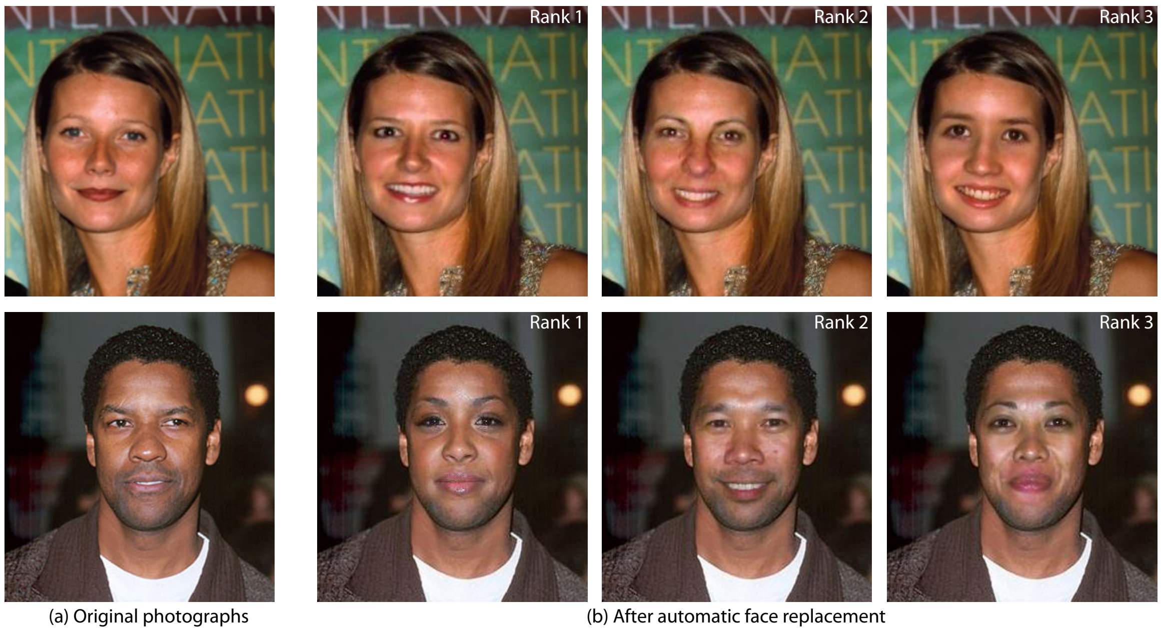


Figure 1.5: We have developed a system that automatically replaces faces in an input image with ones selected from a large collection of face images. Candidate replacements that match a variety of attributes are chosen so that the replacement problem becomes much easier. In this example, the faces of (a) two people are shown after (b) automatic replacement with the top three ranked candidates. Our system for face replacement can be used for face de-identification, personalized face replacement, and creating an appealing group photograph from a set of "burst" mode images.

for example, everyone smiling and with both eyes open.

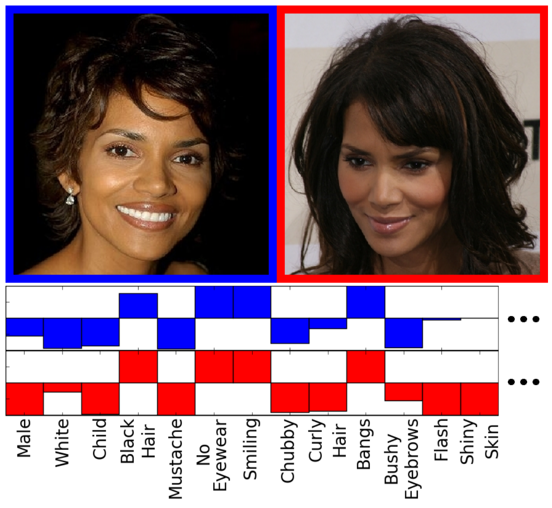
Figure 1.5 shows example results of such a system for fully-automatic face replacement in photographs. We first construct a large library of aligned faces, off-line, once, and which can be efficiently accessed during face replacement. Given an input image (left column), we detect all faces that are present, align them to the coordinate system used by our face library, and select candidate face images from our face library that are similar to the input face in terms of appearance, pose, and other attributes. We then adjust the pose, lighting, and color of the candidate face images to match the appearance of those in the input image, and seamlessly blend in the results. Finally, we rank the blended candidate replacements by computing a match distance over the overlap region. (The top 3 ranked replacements are shown in the remaining columns.) Our approach requires no 3D model, is fully automatic, and generates highly plausible results across a wide range of skin tones, lighting conditions, and viewpoints. As in the search application, the library of faces used for replacement can be chosen appropriately based on the desired application – privacy protection can use a database of stock or synthetic faces, personal photo enhancement can rely on the user's own photo collection, and group shot replacement will, of course, use just the photos taken in a burst.

1.5 Face Verification (Chapter 7)

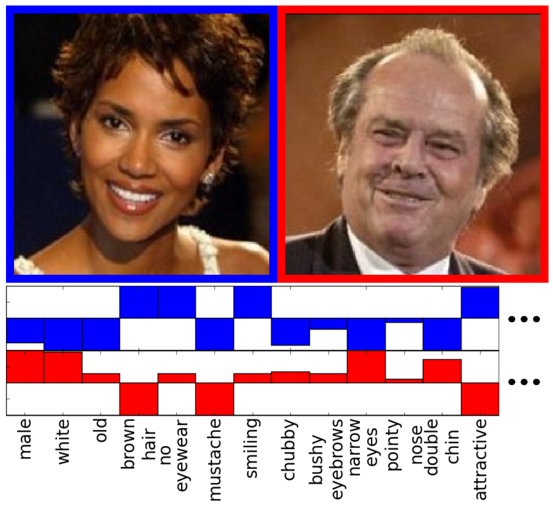
Face verification is the problem of determining whether two faces are of the same individual. It is the fundamental building block for most face recognition algorithms – systems which identify individuals in an image – and is thus an extremely important problem. In security, face recognition can be used to control access to sensitive locations (*e.g.*, secure areas of airports). As another example, there is a need for automatic systems in passport applications to alert humans when a new photo does not appear to depict the same person as did a previous one. In information retrieval, a high percentage of photos on the internet and in personal collections contain faces. The identity and attributes of these faces are a crucial element in their effective retrieval. Face recognition is also critical to building automatic systems, such as household robots, that can interact smoothly with people.

What makes this problem difficult is the enormous variability in the manner in which an individual's face presents itself to a camera: not only might the pose differ, but so might the expression and hairstyle. Making matters worse – at least for researchers in biometrics – is that the illumination direction, camera type, focus, resolution, and image compression are all almost certain to vary as well. These manifold differences in images of the same person have confounded methods for automatic face recognition and verification, often limiting the reliability of automatic algorithms to the domain of more controlled settings with cooperative subjects [Sim *et al.*, 2002; Blanz *et al.*, 2002; Phillips *et al.*, 2006; Gross *et al.*, 2001; Phillips *et al.*, 2000; Samaria and Harter, 1994; Georghiades *et al.*, 2001].

Despite great effort, state-of-the-art methods for identification perform poorly in natural settings with varying pose, illumination, and expression, as evidenced by results on the LFW



(a) Attributes for two images of the same person

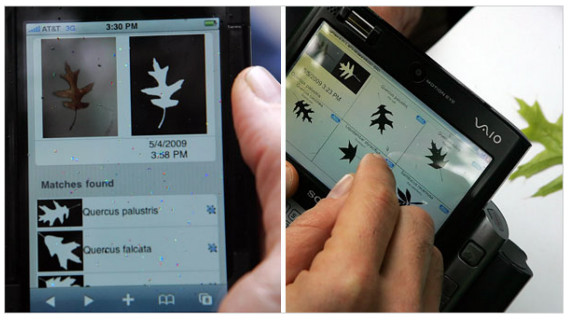


(b) Attributes of two different people

Figure 1.6: An attribute classifier can be trained to recognize the presence or absence of a *describable visual attribute*. The responses of several such attribute classifiers are shown for (a) two images of the same person and (b) two images of different individuals. In (a), notice how most attribute values are in strong agreement, despite the changes in pose, illumination, expression, and image quality. Conversely, in (b), the values differ completely despite the similarity in these same environmental aspects. We train a verification classifier on these outputs to perform face verification, achieving 85.54% accuracy on the Labeled Faces in the Wild (LFW) benchmark [Huang *et al.*, 2007c], comparable to the state-of-the-art.

dataset [Nowak and Jurie, 2007; Wolf *et al.*, 2008; Huang *et al.*, 2007b; Huang *et al.*, 2008; Huang *et al.*, 2007c]. When one analyzes the failure cases for some of these algorithms, many mistakes are found that seem avoidable: men being confused for women, young people for old, asians for caucasians, etc. On the flip side, simple changes in pose, expression, or lighting can cause two otherwise similar images of the same person to be misclassified by an algorithm as different. So while this dataset remains difficult for automatic methods, it is easy for humans: correct verification of identity in image pairs can be performed almost without error [Kumar *et al.*, 2009].

We approach the unconstrained face verification problem with non-cooperative subjects by comparing faces using our attribute and simile classifier outputs, instead of low-level features directly. Fig. 1.6 shows the outputs of various attribute classifiers, for (a) two images



(a)

(b)

Figure 1.7: (a) A prototype of our electronic field guide running as an application for the iPhone and on a Sony Vaio (photo courtesy of the New York Times). (b) Young naturalists using the tablet version of our system at the 2007 BioBlitz, sponsored by the National Geographic Society.

of the same person and (b) images of two different people. Note that in (a), most attribute values are in strong agreement, despite the changes in pose, illumination, and expression, while in (b), the values are almost perfectly contrasting. By training a classifier that uses the difference in these attribute values as inputs for face verification, we achieve close to state-of-the-art performance on the Labeled Faces in the Wild (LFW) data set [Huang *et al.*, 2007c], at 85.54% accuracy. Additionally, we shwo that combining with low-level techniques pushes us past the state-of-the-art to 88.25%.

1.6 Plant Species Identification (Chapter 8)

Finally, leaving the domain of faces and coming back to our motivating example of plants, we describe an automatic plant species identification system. Botanists in the field are racing to capture the complexity of Earth's flora before climate change and development erase their living record. To greatly speed up the process of plant species identification, collection, and monitoring, computational tools are desperately needed by botanists the world over. Without such tools, a dichotomous key must be painfully navigated to search the many branches and seemingly endless nodes of the taxonomic tree. The process of identifying a single species using keys may take hours or days, even for specialists, and is exceedingly difficult or impossible for amateurs.

To bridge this gap, we have built and publicly released – in partnership with the Smithsonian Institution – the first hand-held botanical identification system, *Leafsnap* [Agarwal *et al.*, 2006; Belhumeur *et al.*, 2008]. The system requires only that the user photograph a leaf specimen, and then returns, within seconds, images of the top matching species, along with supporting data such as textual descriptions and high-resolution type specimen images. By using our system, a botanist in the field can now quickly search entire collections of plant species – a process that previously took hours can now be done in seconds. There has been intense public interest in Leafsnap – close to half a million people have downloaded the app for the iPhone and iPad platforms. We have also received an extensive list of requests to use the system by urban planners in local governments, educators throughout the U.S. and abroad, not-for-profit institutions working on issues of biodiversity, and citizen scientists eager to map and monitor the flora of their backyard or local park. Figure 1.7 shows (a) a screenshot from the iPad version of the program and (b) young naturalists using an earlier tablet prototype of our system at the 2007 BioBlitz.

Our current system works by comparing the overall shape of leaves and does a good job at getting the new leaf's species among the top five "hits" more than 90% of the time. However, it often errs in ways that seem avoidable, just as we had previously seen face identification systems err. A leaf may be correctly identified by the system as a Maple – however, the system may be unsure of whether it is a Silver Maple or Sugar Maple, despite the fact that the Silver Maple has a "highly-toothed" margin. Or a Balsam Poplar might be confused for a Swamp Cottonwood, even though the leaf of the Swamp Cottonwood has a "rounded-tip." It is clear to us that there is considerable room and need for improvement by augmenting our system with describable attributes. Furthermore, attributes of the bark, flower, venation or fruit of the plant could be easily combined with attributes of the leaves to better disambiguate similar species. This could perhaps even be done with a human-in-the-loop if needed, an approach currently being taken in the bird identification system currently being developed as part of the Visipedia project [Branson *et al.*, 2010; Welinder *et al.*, 2010; Perona, 2010].

Finally, we conclude in Chapter 9 with some thoughts on the future of attributes in computer vision.

Chapter 2

Background

2.1 Attribute Classification

Prior research on attribute classification has focused mostly on gender and ethnicity classification. Early works [Golomb *et al.*, 1990; Cottrell and Metcalfe, 1990] used neural networks to perform gender classification on small datasets. The Fisherfaces work [Belhumeur *et al.*, 1996] showed that linear discriminant analysis could be used for simple attribute classification such as glasses/no glasses. Later, Moghaddam and Yang [Moghaddam and Yang, 2002] used Support Vector Machines (SVMs) [Cortes and Vapnik, 1995] trained on small "face-prints" to classify the gender of a face, showing good results on the FERET face database [Phillips *et al.*, 2000]. The works of Shakhnarovich *et al.* [Shakhnarovich *et al.*, 2002] and Baluja and Rowley [Baluja and Rowley, 2007] used Adaboost [Freund and Shapire, 1996] to select a linear combination of weak classifiers, allowing for almost real-time classification of face attributes, with results in the latter case again demonstrated on the FERET database. These methods differ in their choice of weak classifiers: the former uses the Haar-like features of the Viola-Jones face detector [Viola and Jones, 2001], while the latter uses simple pixel comparison operators.

In computer vision, the use of attributes has recently been receiving much attention from a number of different groups. The first paper to take a generalized approach to attributes was by Ferrari and Zisserman [Ferrari and Zisserman, 2007], which described a probabalistic approach for learning simple attributes such as colors and stripes. This thesis builds on earlier works [Bitouk *et al.*, 2008; Kumar *et al.*, 2008; Kumar *et al.*, 2009], which were among the first in the community to look at attributes for a variety of computer vision tasks (albeit limited to faces). Other contemporaneous works that use attributes to describe objects include [Lampert *et al.*, 2009], for animal categorization, and [Farhadi *et al.*, 2009], for building general attribute predictors. However, the focus of all of these papers is quite different. The latter [Farhadi *et al.*, 2009] explores how to train attribute classifiers in a very general setting (such as for evaluation on the Pascal VOC challenge [Everingham *et al.*,]) and the problems associated with, *e.g.*, correlations in training data. The former [Lampert *et al.*, 2009], on the other hand, focuses on trying to distinguish animal species and transfer labels across categories. In contrast to both of these approaches, which are trying to find relations across different categories, we concentrate on finding relations between objects in a single category: faces.

Faces have many advantages compared to generic object categories. There is a well-established and consistent reference frame to use for aligning images; differentiating objects is conceptually simple (*e.g.*, it's unclear whether two cars of the same model should be considered the same object or not, whereas no such difficulty exists for two faces); and most attributes can be shared across all people (unlike, *e.g.*, "4-legged," "gothic," or "dual-exhaust," which are applicable to animals, architecture, and automobiles, respectively – but not to each other). All of these benefits make it possible for us to train more reliable and useful classifiers, and demonstrate results comparable to the state-of-the-art.

In psychology and neuroscience, there have been a number of works on face recognition as done by humans. The work of Bruce *et al.* [Bruce *et al.*, 1999] addresses many aspects of human recognition of faces in video and images, including results showing that people are very robust to decreased resolution when recognizing familiar faces, and that the face itself is more useful than the body or gait in such settings [Burton *et al.*, 1999]. In contrast, Sinha and Poggio [Sinha and Poggio, 1996] show an example where context dominates image information in the face region itself. In later work, Sinha *et al.* [Sinha *et al.*, 2006] provide a wide-ranging overview of results from psychology on face recognition, briefly covering the work of Bruce *et al.* [Bruce *et al.*, 1999] and also discussing the effects of varying many other imaging conditions.

Exciting recent work [Palatucci *et al.*, 2009] considers explicitly training attribute classifiers for words, in order to decode fMRI measurements of brain activity while subjects think about words. This work includes initial PAC-style bounds on attribute-based zero-shot learning.

2.2 Content-Based Image Retrieval (CBIR)

Our search application can be viewed as a form of CBIR, where our content is limited to images with faces. Interested readers can refer to the works of Rui *et al*. [Rui *et al.*, 1999], Datta *et al*. [Datta *et al.*, 2005], and Lew *et al*. [Lew *et al.*, 2006] for recent surveys of this field. Some landmark early works included the QBIC [Flickner *et al.*, 1995], PicHunter [Cox *et al.*, 2000], and VisualSEEk [Smith and Chang, 1996] systems. Most relevant to our work is the "Photobook" system [Pentland *et al.*, 1996], which allows for similarity-based searches of faces and objects using parametric eigenspaces. However, their goal is different from ours. Whereas they try to find objects similar to a chosen one, we locate a set of images starting only with simple text queries. Although we use vastly different classifiers and methods for feature selection, their division of the face into functional parts such as the eyes, nose, *etc.*, is echoed in our approach of training classifiers on functional face regions. While in this paper we ignore existing text annotations for images, one could envision using describable attributes in combination with such annotations for improved search performance, somewhat akin to the idea presented in the "Names and Faces" work [Berg *et al.*, 2004].

In the multimedia retrieval community, the use of attributes has become increasingly popular under the term "high-level semantic concepts." Early works explored the feasibility of using concepts for retrieval, reranking, and query expansion, while later works have focused more on how to scale the number of concepts to the thousands, and apply fusion techniques to combine the outputs of multiple concept detectors or multiple modalities into a consistent set of results [Naphade and Smith, 2004; Snoek *et al.*, 2005; Chang *et al.*, 2005; yong Neo *et al.*, 2006; Chang *et al.*, 2006; Naphade *et al.*, 2006; Snoek *et al.*, 2007; Chang *et al.*, 2007; Jiang *et al.*, 2008a; Jiang *et al.*, 2008b; Jiang *et al.*, 2009a; Jiang *et al.*, 2009b; Kennedy and Chang, 2010]. Many of these works have been driven forward through the

TRECVID and LSCOM [Naphade *et al.*, 2006; Yanagawa *et al.*, 2007] projects.

2.3 Face Replacement

While there exists a rich body of work on replacing parts of images with new image data, the replacement of faces in images has received relatively little attention. To the best of our knowledge, the work of [Blanz *et al.*, 2004] is the only published approach which allows one to replace faces in photographs. They fit a morphable 3D model to both the input and target face images by estimating shape, pose and the direction of the illumination. The 3D face reconstructed from the input image is rendered using pose and illumination parameters obtained from the target image. The major drawback of this approach is that it requires manual initialization in order to obtain accurate alignment between the morphable model and the faces in both the input and target images. Although this is acceptable for their goal (virtual hairstyle try-on), our de-identification application absolutely requires that there be no user intervention. A commercial system that also uses 3D models is currently in development at XiD Technologies [1], but details regarding their technical approach, the degree of automation, and the quality of their results are not known. (The estimation of 3D face shape from a single image is an inherently under-constrained problem and by nature difficult to fully automate.) In contrast, our approach allows us to automatically replace faces across different pose and lighting conditions without resorting to 3D methods.

An unpublished work [Malik, 2003] describes another 3D model-based approach for face replacement. This work focuses on improved relighting and recoloring using histogram matching and color blending. However, this system requires manual face alignment to the 3D model, and some post-processing is needed to improve the visual quality of the results. Finally, [Liu *et al.*, 2001] addresses the related, but slightly different problem of transferring expressions of faces between two images. This work introduces a relighting technique which is similar to the one we use for our system, described in Chapter 6.

[1] `http://www.xidtech.com/`

2.3.1 Face De-Identification

The easiest and most well-known method for face de-identification is to distort the image either through pixelation or blur [Boyle *et al.*, 2000], although the results in this case are not as visually appealing. In [Newton *et al.*, 2005], a face de-identification algorithm is introduced which minimizes the probability of automatic face recognition while preserving details of the face. However, since this technique uses Principal Component Analysis to compute averages of images, the replaced faces contain blurring and ghosting artifacts. The work of [Gross *et al.*, 2006] improves the quality of de-identified faces using Active Appearance Models [Cootes *et al.*, 2001] but still suffers from blurring artifacts. Moreover, all of these face de-identification methods work only on faces in frontal pose, and produce images inside a pre-defined face region without any guarantee that the de-identified face will blend well with the rest of the original photograph. Our work differs in that we automatically select faces which yield realistic final results for input faces in different poses, and we perform critical appearance adjustments to create a seamless composite image.

2.3.2 Image Compositing

Some of the applications we describe can be addressed using image compositing approaches. For example, the Photomontage framework [Agarwala *et al.*, 2004] allows a user to interactively create a composite image by combining faces or face parts taken from several source photographs. In [Wang *et al.*, 2007a], faces are replaced using gradient domain image blending. Another approach which uses a large image library as we do is [Hays and Efros, 2007], in which images can be "completed" using elements taken from similar scenes. These image compositing methods are not specifically targeted to face images, however, and they require user interaction to create plausible replacement results. In contrast, our algorithm focuses on faces and can automatically generate a ranked set of replacement results (allowing users to select among them, if desired).

2.4 Face Verification

Early work in appearance-based face verification [Kirby and Sirovich, 1990; Turk and Pentland, 1991] looked at the L_2 distance between pairs of images in a lower dimensional subspace obtained using Principal Components Analysis (PCA). This was extended and improved upon by using linear discriminant analysis [Belhumeur *et al.*, 1996]. However, these algorithms are mostly limited to images taken in highly controlled environments with extremely cooperative subjects. It is well understood that variation in pose and expression and, to a lesser extent, lighting cause significant difficulties for recognizing the identity of a person [Zhao *et al.*, 2003]. Illumination changes can be mostly handled using a variety of different approaches; the direction of the image gradient [Chen *et al.*, 2000b] and related image features such as SIFT [Lowe, 2003], the phase of Gabor jets [Wiskott *et al.*, 1997], and gradient pyramids [Ling *et al.*, 2007] are all highly insensitive to lighting variation. The CMU Pose, Illumination, and Expression (PIE) data set and follow-on results showed that sometimes alignment, especially in 3D, can overcome the other difficulties [Sim *et al.*, 2002; Blanz *et al.*, 2002; Castillo and Jacobs, 2007; Gross *et al.*, 2001; Cootes *et al.*, 2000].

Unfortunately, in the setting of real-world images such as those in the "Labeled Faces in the Wild" (LFW) benchmark data set [Huang *et al.*, 2007c] and similar data sets [Berg *et al.*, 2004; Everingham *et al.*, 2006], 3D alignment is difficult and has not (yet) been successfully demonstrated. Various 2D alignment strategies have been applied to LFW – aligning all faces [Huang *et al.*, 2007b] to each other, or aligning each pair of images to be considered for verification [Nowak and Jurie, 2007; Ferencz *et al.*, 2007; Hua and Akbarzadeh, 2009]. Approaches that require alignment between each image pair are computationally expensive (the latter of these [Hua and Akbarzadeh, 2009] actually aligns images to a common coordinate system, but then computes part-wise similarity between two faces, which requires finding the minimum distance between the features from each part on one face to the best location in the other). Our work does not require pairwise alignment. Neither do many other recent methods on LFW [Pinto *et al.*, 2009; Wolf *et al.*, 2008; Taigman *et al.*, 2009; Wolf *et al.*, 2009], all of which use a large set of carefully designed local features. The best-performing of these [Wolf *et al.*, 2009] ranks the similarity of each face in an input pair to those in a "background set," which is similar in spirit to our simile classifiers.

[More recent LFW results]

Our low-level features are designed following a great deal of work in face recognition (and the larger recognition community) which has identified gradient direction and local descriptors around fiducial features as effective first steps toward dealing with illumination [Chen *et al.*, 2000a; Pentland *et al.*, 1994; Ling *et al.*, 2007; Lowe, 2003; Everingham *et al.*, 2006]. Gallagher and Chen [Gallagher and Chen, 2008] use estimates of age and gender to compute the likelihood of first names being associated with a particular face, but to our knowledge, no previous work has used attributes as features for face verification.

2.5 Plant Species Identification

To our knowledge, no previous work looked at automatic identification for plants. Our approach uses shape retrieval techniques to identify plants based on their contour, and uses histograms of curvatures as the low-level features for training attributes on. We describe previous work in both of these areas.

2.5.1 Shape Retrieval

Curvature has long been used as a basic feature for 2D shape retrieval, but most prior methods computed them using differential methods, which are more sensitive to noise. Unlike us, previous methods either don't use histograms [Mokhtarian *et al.*, 1996; Manay *et al.*, 2006; Koenderink and van Doorn, 1992], or only consider curvatures at a single scale [Manay *et al.*, 2006]. Recent methods for shape retrieval [Yang *et al.*, 2008; Yang *et al.*, 2009] are mostly based on using the Inner Distance Shape Context (IDSC) [Ling and Jacobs, 2007], itself an extension of the Shape Context [Belongie *et al.*, 2002]. IDSC looks at a histogram of distances and angles of contour points to a central point, following the inner-contour. This allows it to handle (small) articulations where necessary, but computational issues make recovering fine-scale details difficult. The latest methods build upon this feature, either by generalizing it [Ling *et al.*, 2010] and/or going beyond pairwise matching to improve results [Ling *et al.*, 2010; Kontschieder *et al.*, 2009]. In our work, we limit ourselves to improving the features and using attribute classifiers. We leave the learning aspects for future work.

2.5.2 Curvature Features

Most relevant to our work is the development of integral measures for computing curvatures, first proposed by Connolly in 1986 [Connolly, 1986]. This work showed how to use the solid angle subtended by an object on a sphere to get a measure of the curvature at a point, in the domain of protein identification. In the vision community, integral measures were first introduced by Manay *et al*. [Manay *et al*., 2006]. They proposed three integral measures in 2D (of which we use two) and gave approximations for their relations to the curvature at a point. While their target application was also 2D shape retrieval, the features they generated were at a single scale and were direct vectorizations of curvature values along a contour. This meant that they also had to solve the alignment problem when comparing images, resulting in lower accuracy (particularly in the presence of articulation or segmentation artifacts) and a much slower algorithm (exhaustively trying all alignments or using the expensive Hausdorff distance). It's also unclear how their method would be applied to images with multiple detected contours.

Our use of histograms over scale is also similar, in spirit, to the multiresolution histograms introduced by Hadjimetriou, *et al*. [Hadjidemetriou *et al*., 2004]. Both approaches use a one-parameter family of operations to generate different histograms, which can be concatenated together to make a more discriminative feature. In our case, this parameter is the scale, implemented in our integral measures by simply changing the radius of the disk or sphere over which we integrate. This has the effect of ignoring perturbations much smaller than the radius. In contrast, multiresolution histograms are obtained by repeatedly smoothing an image, thus incorporating some spatial information by averaging nearby pixels.

Chapter 3

Created Labeled Image Datasets

Two recent trends in internet services have made collecting and labeling image datasets dramatically easier. First, large internet photo-sharing sites such as `flickr.com` and `picasa.com` are growing exponentially and host billions of public images, some with textual annotations and comments. In addition, search engines such as Google Images allow searching for images of particular people (albeit not perfectly). Second, efficient marketplaces for online labor, such as Amazon's Mechanical Turk (MTurk) [Amazon, 2011], make it possible to label thousands of images easily and with very low overhead. We exploit both of these trends to create large datasets of real-world images with attribute and identity labels. (See Fig. 3.1.)

3.1 Collecting Face Images

We use a variety of online sources for collecting face images, including search engines such as Yahoo Images and photo-sharing websites such as `flickr.com`. Depending on the type of data needed, one can either search for particular people's names (to build a dataset labeled by identity) or for default image filenames assigned by digital cameras (to use for labeling with attributes). The latter technique allows one to find images that are otherwise not returned in most users' queries, *i.e.*, images which are effectively "invisible." Relevant metadata such as image and page URLs are stored in the EXIF tags of the downloaded images.

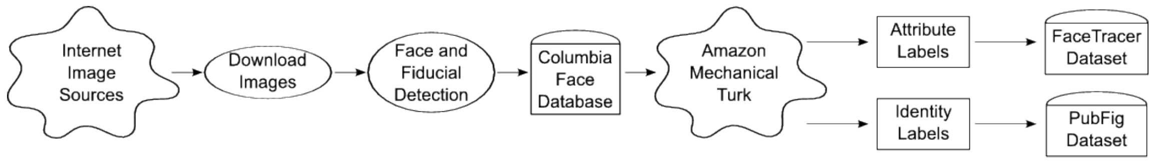


Figure 3.1: Creating labeled image datasets. Our system downloads images from the internet. These images span many sources of variability, including pose, illumination, expression, cameras, and environment. Next, faces and fiducial points are detected using a commercial detector [Omron, 2010] and stored in the Columbia Face Database. A subset of these faces are submitted to the Amazon Mechanical Turk service, where they are labeled with attributes or identity, which are used to create the FaceTracer and PubFig datasets, respectively. Both datasets have been publicly released for non-commercial use.

Next, we apply the OKAO face detector [Omron, 2010] to the downloaded images to extract faces. This detector also returns the 3D pose angles of each face (yaw, pitch, and roll), as well as the locations of six fiducial points: the corners of both eyes and the corners of the mouth. These fiducial points are used to align faces to a canonical frontal pose (center of Fig. 3.2), via an affine transformation computed using linear least squares on the detected points and corresponding points defined on a template face. The points on the template face only have to be defined once, after which they can be automatically used for all future alignments. The alignment procedure takes advantage of the fact that all faces have common structure – *i.e.*, two eyes, a nose, a mouth, *etc.*– and that we have fiducial point detections available from a face detector. The points for faces in different poses are shown in Fig. 3.2.

The distribution of face sizes, as measured by the number of pixels in the cropped face box returned from the detector, are shown in Fig. 3.3a. Note that although most faces are fairly small, there are still a fair number of very large faces as well. Figs. 3.3b and 3.3c show the distribution of yaw and pitch angles, respectively, of faces in our database. Due to both the natural bias of people being photographed frontally and the bias induced by the face detector used, both are centered around 0 degrees (frontal), but still contain main faces out to around 20 degrees or more.

For automatic face replacement, we also assign each face into one of 15 pose bins using its out-of-plane rotation angles. Since the current version of the OKAO detector is less

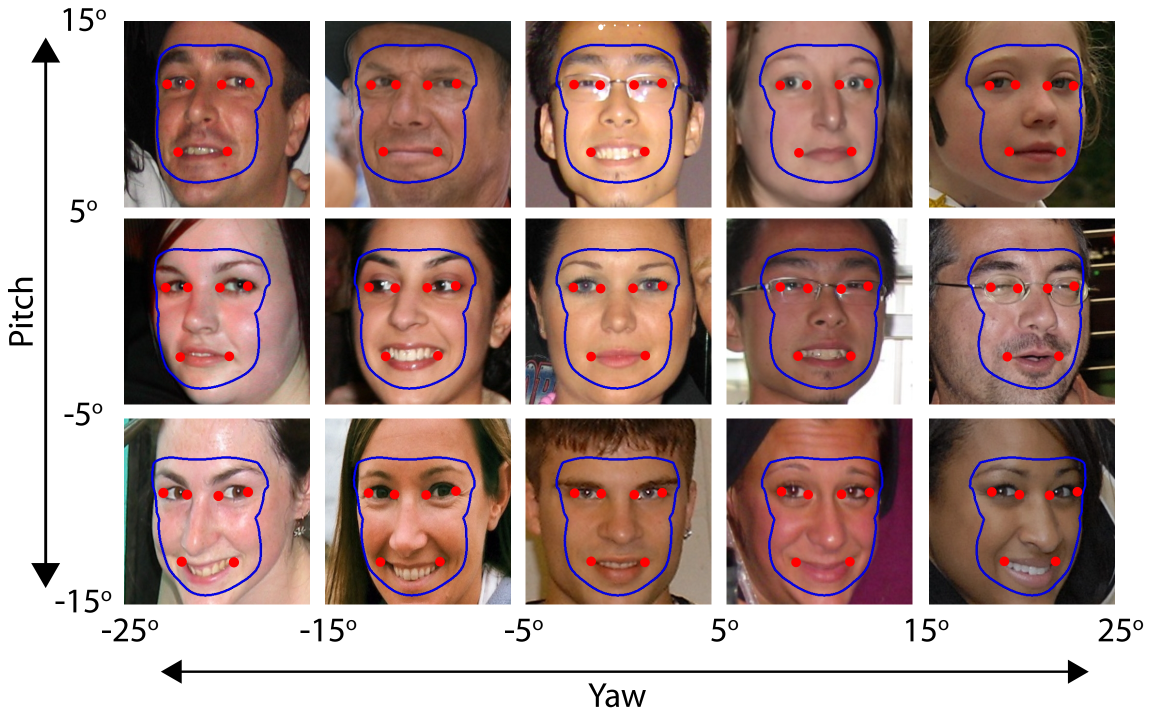
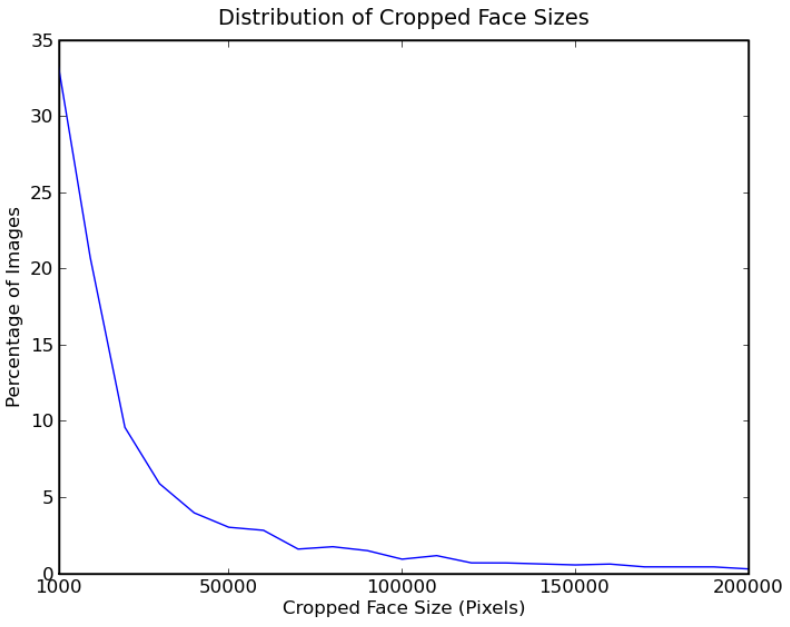


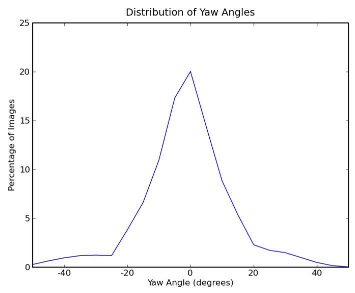
Figure 3.2: The six fiducial points (red) and the outline of the replacement regions (blue) for each of the pose bins. For attribute classification, we align everything to the center bin (frontal view) using an affine alignment. For face replacement, which is more sensitive to alignment artifacts, we align to the closest pose bin. Note that these templates are all defined in 2D, and require no 3D geometry.

reliable for extreme poses, we restricted ourselves to poses within ±25° in yaw and ±15° in pitch. The pose bins, shown in Fig. 3.2, span intervals of 10° from -25° to 25° for the yaw and from -15° to 15° for the pitch. Since face replacement is more sensitive than attribute classification, we define a separate generic face oriented at the center of each pose bin. Since all faces within a bin are guaranteed to have fairly similar poses, we introduce fewer artifacts than 3D methods that attempt to reconstruct faces across very different poses. The bins are shown in Fig. 3.2. Finally, we also add the mirror image of each face, so as to increase the number of candidates in our library.

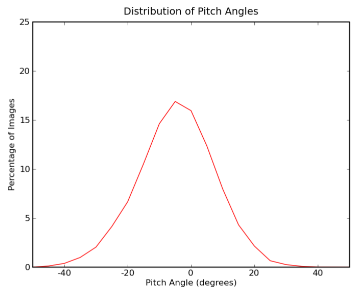
The 3.1 million aligned faces collected using this procedure comprise the Columbia Face Database. Various statistics about this database are presented in Table 3.1. We make two



(a) Distribution of face sizes



(b) Distribution of yaw angles



(c) Distribution of pitch angles

Figure 3.3: (a) The distribution of face sizes in our database. The face size is measured as number of pixels in the cropped face box returned from the face detector. Note that although most faces are fairly small, there are still quite a few large faces as well. (b) and (c) show the distribution of yaw and pitch angles, respectively, of faces in our database. Due to both the natural bias of people being photographed frontally and the bias induced by the face detector used, both are centered around frontal views, but still contain many faces out to around 20 degrees or more.

observations about this database. First, from the statistics of the randomly-named images, it appears that a significant fraction of them contain faces (25.7%), and on average, each image contains 0.5 faces. Thus, it is clear that faces are ubiquitous and an important case to understand. Second, our collection of aligned faces is the largest such collection of which we are aware. It is truly a "real-world" dataset, with completely uncontrolled lighting and environments, taken using unknown cameras and in unknown imaging conditions, with a wide range of image resolutions. In contrast, existing face datasets such as Yale Face A&B [Georghiades *et al.*, 2001], CMU PIE [Sim *et al.*, 2002], and FERET [Phillips *et al.*, 2000] are either much smaller in size and/or taken in highly controlled settings. Even the more expansive FRGC version 2.0 dataset [Phillips *et al.*, 2005] has a limited number

Image Source	**# Images Down-loaded**	**# Images With Faces**	**% Images With Faces**	**Total # Faces Found**	**Average # Faces Found Per Image**
Randomly Downloaded	4,289,184	1,102,964	25.715	2,156,287	0.503
Celebrities	428,312	411,349	96.040	285,627	0.667
Person Names	17,748	7,086	39.926	10,086	0.568
Face-Related Words	13,028	5,837	44.804	14,424	1.107
Event-Related Words	1,658	997	60.133	1,335	0.805
Professions	148,782	75,105	50.480	79,992	0.538
Series	7,472	3,950	52.864	8,585	1.149
Camera Defaults	895,454	893,822	99.818	380,682	0.425
Miscellanous	417,823	403,233	96.508	194,057	0.464
Total	**6,219,461**	**2,904,343**	**46.698**	**3,131,075**	**0.503**

Table 3.1: Image database statistics. We have collected what we believe to be the largest set of aligned real-world face images (over 3.1 million so far). These faces have been extracted using a commercial face detector [Omron, 2010]. Notice that more than 45% of the downloaded images contain faces, and on average, there is one face per two images.

of subjects, image acquisition locations, and all images were taken with the same camera type. The most comparable dataset is LFW [Huang *et al.*, 2007c], itself derived from earlier work [Berg *et al.*, 2004]. These images were collected from news sources, and exhibit many of the same types of variation as the Columbia Face Dataset.

3.2 Collecting Attribute and Identity Labels

For labeling images in our Columbia Face Database, we use the Amazon Mechanical Turk (MTurk) service [Amazon, 2011]. On MTurk, "requesters" can submit jobs to be completed by workers, optionally setting various quality controls such as confirmation of results by multiple workers, filters on minimum worker experience, *etc.* The jobs we created typically took between 30 seconds and a few minutes, and paid between $0.01 to $0.10. Jobs included data cleanup, online research, and manual annotation, among many others. More details on using MTurk effectively can be found in Appendix A.

We submitted 110,000 attribute labeling jobs showing 30 images to 3 workers per job, presenting a total of over 10 million images to users. The jobs asked workers to select face images which exhibited a specified attribute. (A few manually-labeled images were shown as examples.) Only labels where all 3 people agreed were used. From this raw data, we were able to collect over 145,000 triply-verified positive and 1,950,000 negative attribute labels, respectively, for about $6,000. Table 3.2 lists the attributes and the number of confirmed positive and negative labels for each.

Although this approach is somewhat similar to other labeling efforts in the computer vision community – such as ImageNet [Deng *et al.*, 2009] and LabelMe [Russell *et al.*, 2008], which focus on naming objects, images, and regions of images using nouns – there are several important differences. One is that attributes need not be binary or even discrete; a person's age or the thickness of their eyebrows are both continuous attributes. (However, in this work we only consider discrete attributes, to simplify labeling.) Another critical difference is that visual attributes can be composed more freely than names, which generally exist in a tree-structured hierarchy. This allows for the use of a set of general attributes, which can be combined in an exponential number of ways to describe many objects at different levels of specificity. Attributes can therefore compactly provide a great deal of information, both about object properties and their identity. Finally, for many objects, it can be prohibitively expensive to obtain a large number of labeled training images of a specific object or category. In contrast, the same attribute can be exhibited by many otherwise-unrelated objects, making it easier to find more training images.

For gathering identity labels, we used the images downloaded from keyword searches on people's names as raw inputs, which were then filtered to create the final set. We submitted MTurk jobs asking users to select only the face images of a given person (of whom a few examples were shown). We also ran additional jobs pruning images for quality, good alignment, and some conservative duplicate-removal.

From these attribute and identity labels and our face database, we have created two publicly available face datasets, described next.

3.2.1 FaceTracer Dataset

The FaceTracer dataset is a subset of the Columbia Face Database, and it includes attribute labels. Each of the $15,000$ faces in the dataset has a variety of metadata and fiducial points marked. The attributes labeled include demographic information such as age and race, facial features like mustaches and hair color, and other attributes such as expression, environment, *etc.* There are $5,000$ labels in all. FaceTracer can be used as simply a dataset of real-world images with face detections and fiducials; or by researchers wanting to train their own attribute classifiers; or for any other non-commercial purpose.

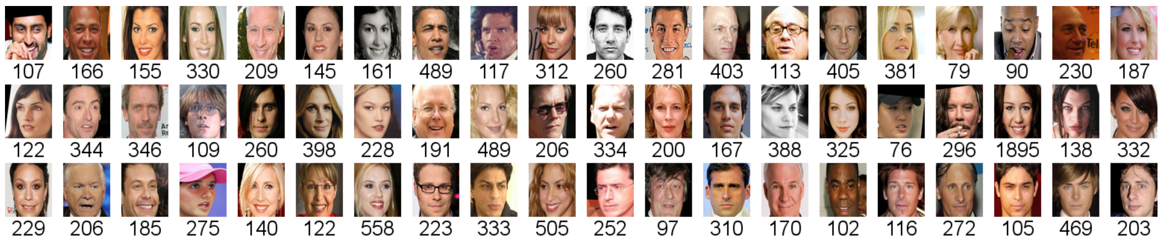
The dataset is publicly available as a set of face URLs and accompanying data at `http://faceserv.cs.columbia.edu/databases/facetracer/`

3.2.2 PubFig Dataset

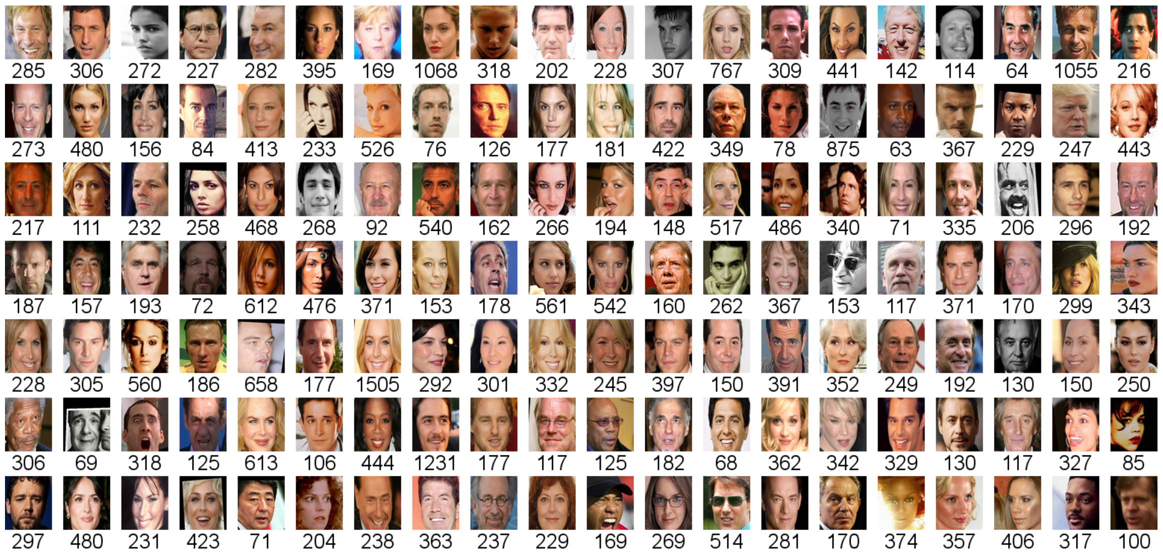
The PubFig dataset is a complement to the LFW dataset [Huang *et al.*, 2007c]. It consists of $58,797$ images of 200 public figures. The larger number of images per person (as compared to LFW) allows one to construct subsets of the data across different poses, lighting conditions, and expressions for further study. Figure 3.4c shows the variation present in all the images of a single individual (Steve Martin). In addition, this dataset is well-suited for recognition experiments.

PubFig is divided into a development set of 60 people (shown in Fig. 3.4a), on which we trained our simile classifiers, and an evaluation set of 140 people (shown in Fig. 3.4b). The evaluation set was used to create a face verification benchmark similar to that from LFW.

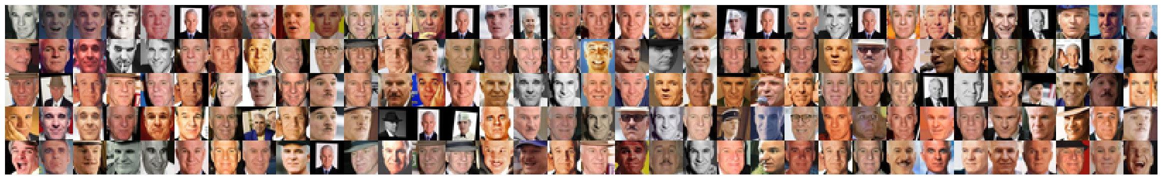
All the data (with URLs to images) and evaluation benchmarks from PubFig are publicly available for non-commercial use at `http://faceserv.cs.columbia.edu/databases/pubfig/`, which also includes information on pose, expression and illumination for the evaluation set, and the outputs of our attribute classifiers on all images in both the development and evaluation sets.



(a) PubFig Development set (60 individuals)



(b) PubFig Evaluation set (140 individuals)



(c) All 170 images of Steve Martin

Figure 3.4: The PubFig dataset consists of 58, 797 images of 200 public figures – celebrities and politicians – partitioned into (a) a development set of 60 individuals and (b) an evaluation set of 140 individuals. Below each thumbnail is shown the number of photos of that person. There is no overlap in either identity or image between the development set and any dataset that we evaluate on, including Labeled Faces in the Wild [Huang *et al.*, 2007c]. The immense variability in appearance captured by PubFig can be seen in (c), which shows all 170 images of Steve Martin.

Attribute	Labels	Attribute	Labels	Attribute	Labels
5 o' Clock Shadow	866	Indian	915	Pale Skin	966
Arched Eyebrows	996	Indoor	1,233	Photo of Face Photo	202
Asian	1,959	Male	2,365	Pointy Nose	1,030
Athletic Shot	192	Middle Aged	1,074	Posed Photo	1,060
Attractive Man	1,000	Mouth Closed	1,190	Receding Hairline	993
Attractive Woman	946	Mouth Obstruction	160	Redhead	980
Baby	980	Mouth Slightly Open	1,160	Rosy Cheeks	990
Bags Under Eyes	995	Mouth Wide Open	991	Round Eyes	1,128
Bald	901	Mustache	982	Round Face	1,092
Bangs	1,066	Narrow Eyes	1,209	Round Jaw	1,193
Big Lips	1,027	No 5 o' Clock Shadow	1,391	Senior	950
Big Nose	1,013	No Bags Under Eyes	1,098	Sharp	1,376
Black	744	No Bangs	1,420	Shiny Skin	982
Black Hair	1,303	No Beard	1,438	Sideburns	880
Black and White	934	No Cleft Chin	1,543	Small Nose	1,047
Blocked Forehead	1,187	No Crow's Feet	1,153	Smiling	1,376
Blond Hair	1,064	No Double Chin	1,437	Soft Lighting	985
Blue Eyes	1,052	No Eyewear	2,794	Square Face	1,021
Blurry	1,192	No Heavy Makeup	1,393	Square Jaw	789
Brown Eyes	1,025	No Mouth Obstruction	1,120	Straight Eyebrows	994
Brown Hair	1,018	No Mustache	2,653	Straight Hair	1,245
Bushy Eyebrows	939	No Necklace	443	Nose-Mouth Lines	798
Casual Photo	1,078	No Nose-Mouth Lines	1,010	Sunglasses	983
Child	1,134	No Sideburns	1,142	Teeth Not Visible	1,156
Chubby	1,006	Not Athletic Shot	1,474	Teeth Visible	1,390
Color Photo	1,438	Not Flushed Face	1,134	Thin	1,248
Crow's Feet	954	Not High Cheekbones	1,018	Thin Eyebrows	1,096
Curly Hair	890	Not Pale Skin	1,148	Thin Lips	1,002
Double Chin	904	Not Photo of Face Photo	1,013	Unattractive Man	993
Eyeglasses	1,132	Not Receding Hairline	1,271	Unattractive Woman	1,093
Eyes Closed	995	Not Rosy Cheeks	1,147	Visible Forehead	1,421
Eyes Open	2,297	Not Shiny Skin	1,140	Wavy Hair	931
Female	2,465	Not Wearing Earrings	1,064	Wearing Earrings	1,062
Flash	1,053	Not Wearing Eye Shadow	660	Wearing Eye Shadow	187
Flushed Face	998	Not Wearing Hat	2,781	Wearing Hat	1,044
Frowning	965	Not Wearing Lipstick	1,390	Wearing Lipstick	962
Full Beard	697	Not Wearing Necklace	1,120	Wearing Necklace	1,030
Goatee	763	Not Wearing Necktie	979	Wearing Necktie	1,030
Gray Hair	960	Not Wrinkled Skin	1,037	White	2,541
Harsh Lighting	989	Obstructed Forehead	1,154	Wide Nose	1,025
Heavy Makeup	1,107	Outdoor	1,064	Wrinkled Skin	206
High Cheekbones	995	Oval Face	1,104	Youth	1,543

Table 3.2: We have collected over 145, 000 triply-verified positive attribute labels for many attributes and hundreds of thousands of images using Mechanical Turk [Amazon, 2011].

Chapter 4

Training Attribute Classifiers

Given a particular describable visual attribute – say "gender" for a face, or "serrated" for a leaf – how can one train an estimator for the attribute? Let us address this problem by first formalizing our notion of attributes. As stated in the previous section, attributes can be thought of as functions $\mathbf{a_i}$ that map instances X to real values a_i. Evaluating the function can be thought of as measuring the attribute. Large positive values of a_i indicate the presence or strength of the ith attribute, while negative values indicate its absence.

Consider the attribute "gender." If images X_1 and X_2 are of males and image Y is of a female, we would like our gender function $\mathbf{a_g}$ to map these input images onto the real line with males assigned positive values and females negative values. Notice that images X_1 and X_2 could differ in many respects – lighting, pose, age, expression, and other attributes – and yet the gender estimator should still mark them as male. We would like these estimators to measure the degree of the attribute as well. For instance, if X_1 were an image of Clint Eastwood and X_2 were an image of Orlando Bloom, we would want $\mathbf{a_g}(X_1) > \mathbf{a_g}(X_2)$. Similarly, the estimator for a different attribute – age, for example – should give reliable results despite changes in gender.

Similes are another class of describable visual traits, which describe the similarity of a face region between two different individuals. For example, we could say a person has "eyes like Penelope Cruz's" or a "mouth like Angelina Jolie's." We can formalize these two simile functions as $\mathbf{s_{cruz_{eyes}}}$ and $\mathbf{s_{jolie_{mouth}}}$; someone who shared Cruz's eyes but not Jolie's mouth would thus have a positive value for the former and a negative value for the latter.

We pose the problem of learning an attribute or simile classifier as one of supervised learning: fitting a function $\mathbf{a_i}$ to a set of labeled training data. If the training labels are ± 1, this can be seen as fitting a classification function; real-valued labels imply regression; and if only ordering constraints are given, it becomes a problem of learning ranking functions. In all cases, regularization is important because the inputs (low-level image features) are very high-dimensional with complex variation, and there is always limited training data. This regularization could be biased by the distribution of features actually observed, which can be acquired from both labeled and unlabeled data. In this work, we consider mainly binary classifiers.

To be useful, we will need several attribute classifiers (tens or hundreds). Thus, it becomes infeasible to manually gather the necessary data, label the training examples, design an appropriate set of relevant low-level features to use for classification, or train the final classifier. Rather, everything must be automated.

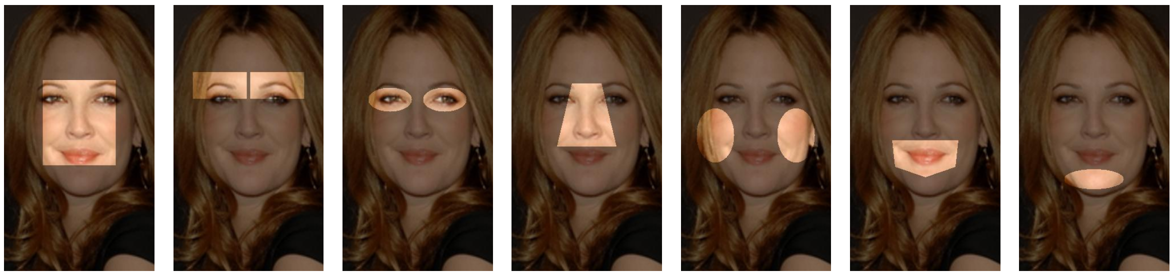
4.1 Low-Level Features

As described in the previous chapter, face images are first aligned using an affine transformation. A set of k low-level feature extractors $\mathbf{f_j}$ are applied to an aligned input image I to form a feature set $\mathcal{F}(I)$:

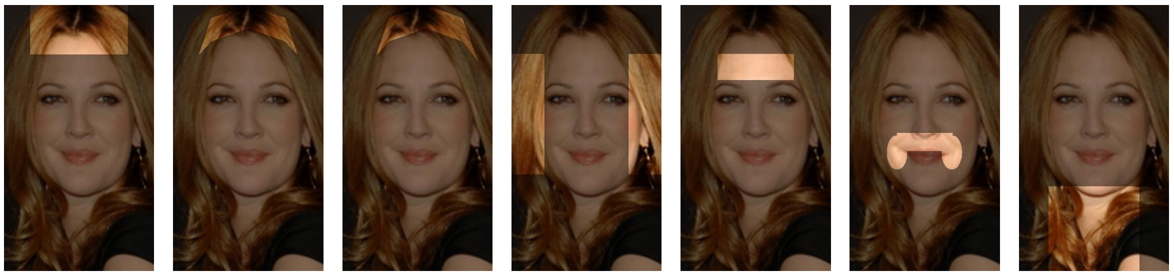
$$\mathcal{F}(I) = \{\mathbf{f_1}(I), \cdots, \mathbf{f_k}(I)\}\,. \tag{4.1}$$

We describe each extractor $\mathbf{f_j}$ in terms of four choices: the region of the face to extract features from, the type of pixel data to use, the kind of normalization to apply to the data, and finally, the level of aggregation to use.

Our complete set of regions are shown in Fig. 4.1. The regions correspond to functional parts of a face, such as the nose, mouth, *etc.*, similar to those defined in the work on modular eigenspaces [Pentland *et al.*, 1994]. Regions are defined manually in the affine-aligned coordinate system. This only has to be done once, after which all aligned faces can use the same region definitions. Our coarse division of the face allows us to take advantage of the common geometry shared by faces, while allowing for differences between individual faces as well as robustness to small errors in alignment. Prior to feature extraction, we



(a) face (b) eyebrows (c) eyes (d) nose (e) cheeks (f) mouth (g) chin



(h) hair (i) hair 2 (j) hair 3 (k) side hair (l) forehead (m) mustache (n) neck

Figure 4.1: The face regions used for automatic feature selection are shown here on an affine-aligned face image. There is (a) one region for the whole face, and (b-n) other regions corresponding to functional parts of the face, such as the mouth, eyes, nose, *etc*. Regions are large enough to contain the face part across changes in pose, small errors in alignment, and differences between individuals. The regions are manually defined, once, in the affine-aligned coordinate system, and can then be used automatically for all aligned input faces.

Pixel Value Types	**Normalizations**	**Aggregation**
RGB	None	None
HSV	Mean Normalization	Histogram
Image Intensity	Energy Normalization	Mean/Variance
Edge Magnitude		
Edge Orientation		

Table 4.1: Feature type options. A complete feature type is constructed by first converting the pixels in a given region (see Fig. 4.1) to one of the pixel value types from the first column, then applying one of the normalizations from the second column, and finally aggregating these values into the output feature vector using one of the options from the last column.

mask out the background to avoid contaminating the classifiers. We also use the detected yaw angles of the face to first flip images so that they always face left. This small tweak makes the classifier's job slightly easier, as the "good" side of the face is always on the same half of the image.

From each region, one can extract different types of information, as categorized in Table 4.1. The types of pixel data to extract include various color spaces (RGB, HSV) as well as edge magnitudes and orientations. To remove lighting effects and better generalize across a limited number of training images, one can optionally normalize these extracted values. One method for normalization is mean normalization, $\hat{x} = \frac{x}{\mu}$, which removes illumination gains. Another option is energy normalization, $\hat{x} = \frac{x-\mu}{\sigma}$, which removes gains as well as offsets. (In these equations, x refers to the input value, μ and σ are the mean and standard deviation of all the x values within the region, and $\hat{x}$ refers to the normalized output value.) Finally, one can aggregate normalized values over the region rather than simply concatenating them. This can be as simple as using only the mean and variance, or include more information by computing a histogram of values over the region. A complete feature type is created by choosing a region from Fig. 4.1 and one entry from each column of Table 4.1. (Of course, not all possible combinations are valid; *e.g.*, it doesn't make sense to normalize hues.)

4.2 Feature Selection and Classifier Training

In creating a classifier for a particular attribute, we could simply extract all types of low-level features from the whole face, and let a classifier figure out which are important for the task and which are not. This, however, puts too great a burden on the classifier, confusing it with non-discriminative features. Instead, we design a selection procedure which automatically chooses the best features from a rich set of feature options. The chosen features are used to train the final attribute or simile classifier.

Our feature selection uses an iterative greedy approach. In each iteration, we train several individual classifiers on the current set of features in the output set, concatenated with a single region-feature combination. Each classifier's performance is evaluated using

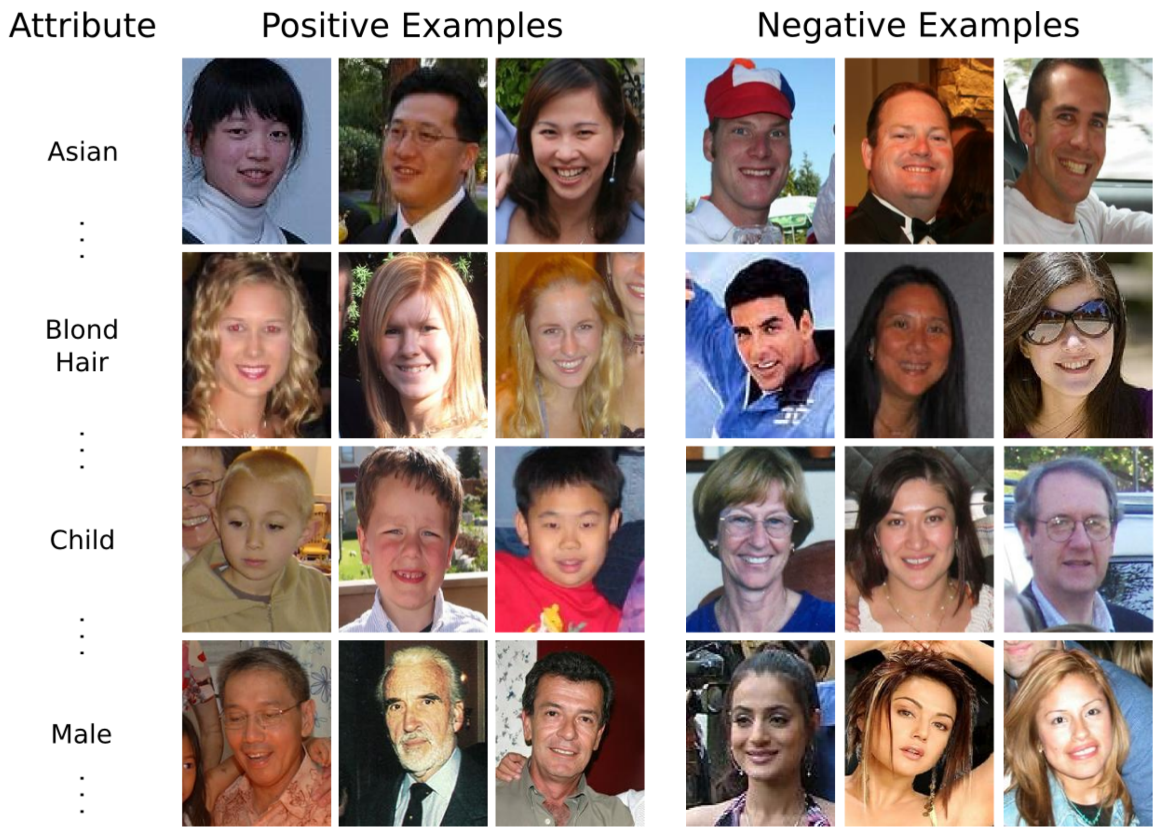


Figure 4.2: Training data for the attribute classifiers consists of face images that match the given attribute label (positive examples) and those that don't (negative examples). Shown here are a few of the training images used for four different attributes. Final classifier accuracies for all 73 attributes are shown in Table 4.2.

cross-validation. The features used in the classifier with the highest cross-validation accuracy are added to the output set. We continue adding features until the accuracy stops improving, up to a maximum of 6 low-level features. For computational reasons, we drop the lowest-scoring 70% of features at each round, but always keeping at least 10 features.

Our classifiers are Support Vector Machines (SVMs) [Cortes and Vapnik, 1995] with RBF kernels, trained using libsvm [Chang and Lin, 2001]. For each classifier, we use between 500 to 2000 positive and negative examples each, and perform a grid search over the C and γ parameters for the SVM. The entire process is fully automatic, and takes a few hours of computation time per attribute trained, using a small grid of roughly 10 Intel Xeon processors, running at 3.0 Ghz each. (Some example timings: "gender" took about 29

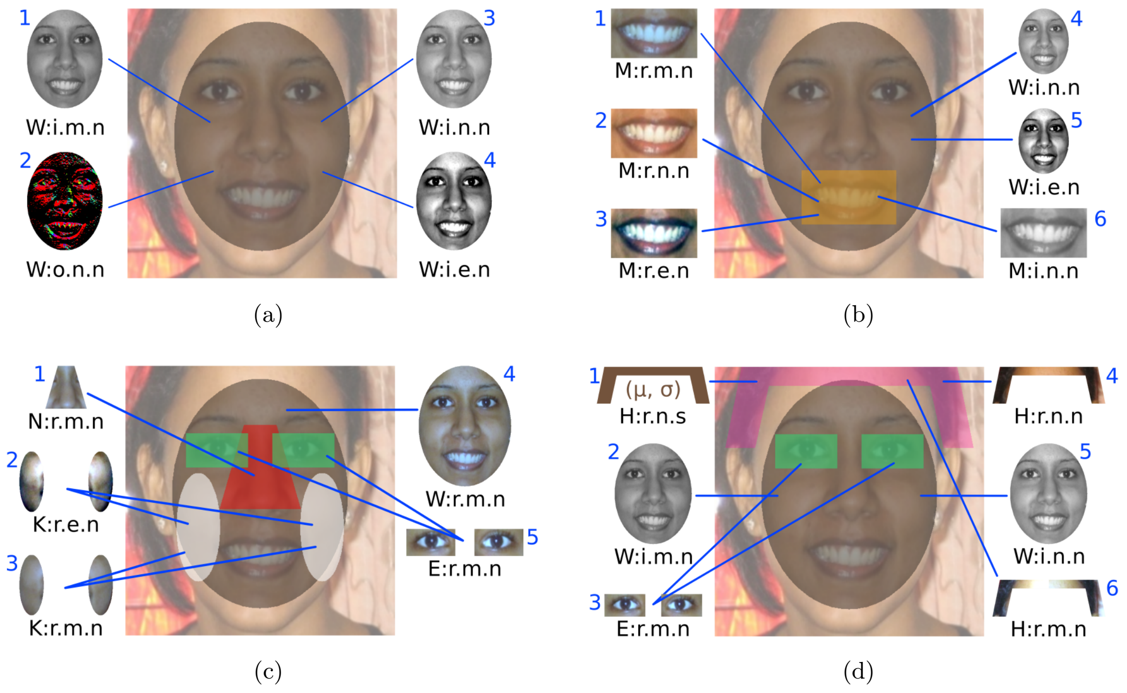


Figure 4.3: Illustrations of automatically-selected region and feature types for (a) gender, (b) smiling, (c) environment, and (d) hair color. Each face image is surrounded by depictions of the top-ranked feature combinations for the given attribute. Notice how each classifier uses different regions and feature types of the face.

hours of total machine-hours, while "sideburns" took about 9.) We note that our procedure is by no means optimal; picking optimal features for non-linear classifiers is still an open problem in machine learning. Nevertheless, we obtain excellent results in practice.

Using the Columbia Face Database and the learning procedure just described, we have trained a total of 73 attribute classifiers. Their cross-validation accuracies are shown in Table 4.2, and typically range from 80% to 90% (random performance would be 50% for each attribute). Analysis of the chosen features indicate that all regions and feature types are useful (to varying extents), suggesting the importance of performing feature selection.

In Fig. 4.3, we visually illustrate the top feature combinations chosen for the gender, smiling, environment, and hair color attributes. This figure shows the ability of our feature selection approach to identify the relevant regions and feature types for each attribute.

Attribute	**Acc.**	**Attribute**	**Acc.**
Gender	85.8%	Nose Size	86.5%
Asian	93.8%	Nose Shape	87.0%
Caucasian	91.5%	Nose-Mouth Lines	93.2%
African American	94.6%	Mustache	92.5%
Indian	91.9%	Mouth Closed	90.0%
Baby	93.0%	Mouth Open	84.6%
Child	80.3%	Mouth Wide Open	89.0%
Youth	87.7%	Lip Thickness	82.4%
Middle-Aged	84.9%	Wearing Lipstick	86.7%
Senior	92.0%	Teeth Visible	91.2%
Black Hair	90.8%	5 o'clock Shadow	89.3%
Blond Hair	88.4%	Beard	88.7%
Brown Hair	74.9%	Goatee	80.4%
Gray Hair	89.9%	Double Chin	81.0%
Bald	90.4%	Jaw Shape	66.1%
Wearing Hat	89.1%	Chubby Face	81.2%
Curly Hair	70.1%	Oval Face	73.3%
Wavy Hair	66.6%	Square Face	78.6%
Straight Hair	78.4%	Round Face	75.5%
Receding Hairline	86.8%	Heavy Makeup	89.0%
Bangs	91.5%	Shiny Skin	84.2%
Visible Forehead	89.3%	Pale Skin	89.4%
Obscured Forehead	77.0%	Flushed Face	88.8%
Blocked Forehead	81.2%	Smiling	95.9%
Eyebrow Thickness	94.6%	Frowning	95.3%
Eyebrow Shape	79.7%	Wearing Necktie	83.7%
Eye Shape	89.7%	Wearing Necklace	67.3%
Eyes Open	92.3%	Blurry Image	93.4%
Eye Color	86.8%	Harsh Lighting	77.0%
No Eyewear	93.3%	Flash Lighting	73.4%
Eyeglasses	92.4%	Soft Lighting	68.5%
Sunglasses	96.5%	Environment	85.3%
Bags Under Eyes	85.4%	Color Photo	97.9%
Wearing Earrings	77.6%	Posed Photo	71.9%
Sideburns	72.3%	Attractive Man	74.2%
High Cheekbones	86.1%	Attractive Woman	82.6%
Rosy Cheeks	86.2%		

Table 4.2: We present accuracies of the 73 attribute classifiers trained using the procedure described in this chapter. Example training images for the attributes in **bold** are shown in Fig. 4.2.

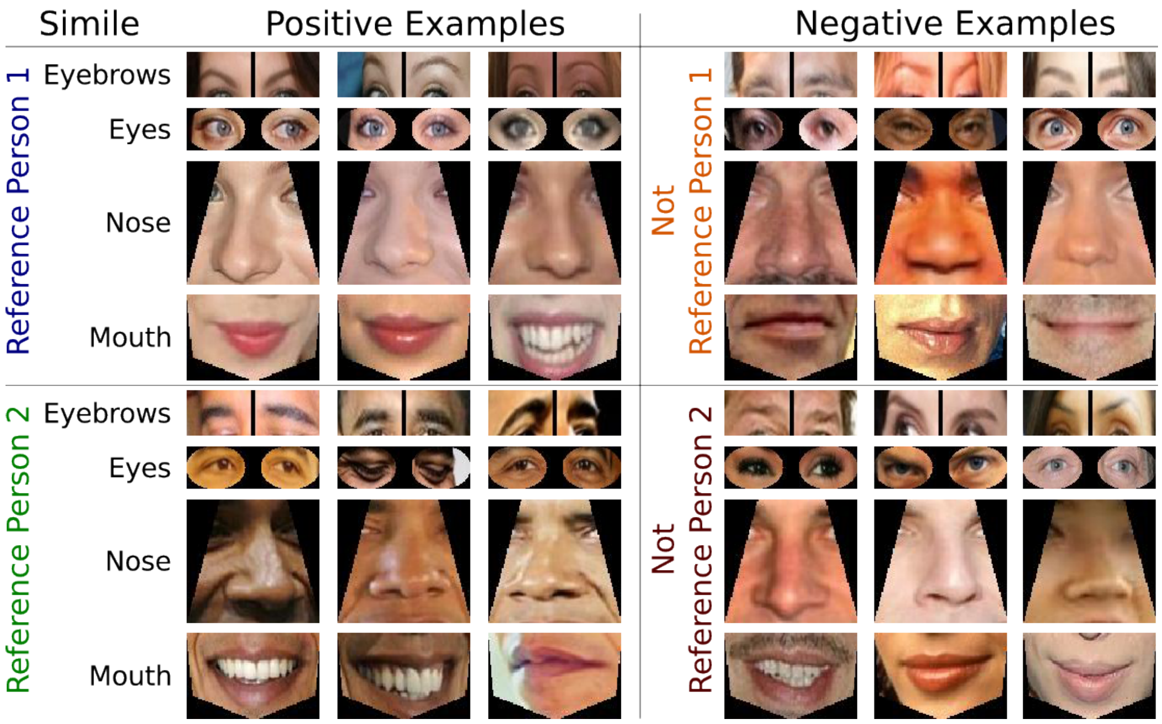


Figure 4.4: Each simile classifier is trained using several images of a specific reference person as the positive data, limited to a small face region such as the eyes, nose, or mouth, and the corresponding regions from images of other people as the negative data. We show here three positive and three negative examples each, for four regions on two of the reference people used to train these classifiers.

4.2.1 Training Simile Classifiers

Simile classifiers measure the similarity of part of a person's face to the same part on a set of reference people. We use the 60 individuals from the development set of PubFig as the reference people. Figure 4.4 shows examples of regions selected from two reference people in the training data. For each reference person in the training set, several simile classifiers are trained for each face region (one per feature type), yielding a large set of total classifiers.

For each reference person, we train support vector machines to distinguish a region (*e.g.*, eyebrows, eyes, nose, mouth) on their face from the same region on other faces. We manually choose eight regions and six feature types from the set of possible features described in Table 4.1 and train classifiers for each reference person/region/feature type combination, without feature selection, yielding $2,880$ total simile classifiers. Each simile classifier is an RBF SVM, trained using at most 600 positive samples of a reference person

	Error Rates	
Classification Method	**Gender**	**Smiling**
Attribute Classifiers	**8.62%**	**4.67%**
Pixel comp. feats. [Baluja and Rowley, 2007]	13.13%	7.41%
Haar-like feats. [Shakhnarovich *et al.*, 2002]	12.88%	6.40%
Full-face SVM [Moghaddam and Yang, 2002]	9.52%	13.54%

Table 4.3: Comparison of attribute classification accuracy for "gender" and "smiling" attributes. Our fully-automatic feature selection and training procedure learns better classifiers than prior state-of-the-art methods for both attributes. (For this comparison, classifiers were trained and evaluated using only near-frontal faces.)

and at most 10 times as many negative samples, randomly chosen from images of other people in the training set.

We emphasize two points. First, the individuals chosen as reference people *do not appear* in LFW or other benchmarks on which we produce results. Second, we train simile classifiers to recognize similarity to *part* of a reference person's face in *many* images, not similarity to a single image. The use of face parts increases the number of classifiers, but makes each one easier to learn, while the use of several input images allows for much better generalizability.

4.3 Comparisons to Prior Work

While we have designed our classifier architecture to be flexible enough to handle a large variety of attributes, it is important to ensure that we have not sacrificed accuracy in the process. We therefore compare our approach to three previous state-of-the-art methods for attribute classification: full-face SVMs using brightness normalized pixel values [Moghaddam and Yang, 2002], Adaboost using Haar-like features [Shakhnarovich *et al.*, 2002], and Adaboost using pixel comparison features [Baluja and Rowley, 2007]. Since these works have mostly focused on gender classification, we use that attribute as the first testing criteria. In addition, we also test performance on the "smiling" attribute – which we expect to be localizable to a small region of the face: the mouth.

Results are shown in Table 4.3. Our method performs the best in all cases (in some cases significantly so). This highlights the power of doing feature selection; in particular, we see that the full-face SVM method, while performing reasonably well on gender, did much worse on a localized attribute like smiling. Note that for the purposes of this test, we limited training and evaluation images to mostly frontal faces.

Chapter 5

Face Search

With the ubiquity of face images on the internet and in people's private collections, new tools are needed for searching and exploring these ever-increasing image datasets and video streams. The possible applications for image search are almost endless, even when limited to images of faces. As just a few examples: A researcher for ABC World News might search a video archive for all clips of President Nixon "laughing." A mother might browse her personal photo collection for a holiday card photo, limiting the search by typing "kids eyes-open smiling." Also, one could augment attribute searches with images. For example, a law enforcement officer might upload an image of a suspect and search for images of him with no beard and glasses.

The ability of current search engines to find images based on facial appearance is limited to images with text annotations. Yet, there are many problems with annotation-based search of images: the manual labeling of images is time-consuming; the annotations are often incorrect or misleading, as they may refer to other content on a webpage; and finally, the vast majority of images are simply not annotated. As mentioned in the introductory chapter, Fig. 1.4 shows the results of the query, "smiling asian men with glasses," using both a conventional image search engine and an attribute-based one. The conventional search engine's reliance on text annotations results in it finding images that have no relevance to the query. In contrast, our prototype **FaceTracer** system handles this same query much better – returning only relevant results.

The FaceTracer engine uses simple text-based queries as inputs, since these are both familiar and accessible to most internet users, and correspond well to describable visual attributes. Search queries are mapped onto attribute labels using a dictionary of terms. Users can see the list of attributes supported by the system on the search page, allowing them to construct searches without having to guess what kinds of queries are allowed. This approach is simple, flexible, and yields excellent results in practice. Furthermore, it is easy to add new phrases and attributes to the dictionary, or maintain separate dictionaries for searches in different languages.

5.1 Details

Like much of the work in content-based image retrieval, the power of our approach comes from automatically labeling images off-line on the basis of a large number of attributes. At search time, only these labels need to be queried, resulting in almost instantaneous searches. Furthermore, it is easy to add new images and face attributes to our search engine, allowing for future scalability. Defining new attributes and manually labeling faces to match those attributes can also be done collaboratively by a community of users.

The search problem is simple to state: a user enters one or more query terms, and the system returns results ranked by relevance. An input query Q, consisting of text provided by the user, is parsed using a language parser $p(Q)$. This parser takes the text and converts it into a set of weighted attributes. For example, the query "male" might translate to the attribute set $\{male = 1\}$, whereas a query "boy" might translate to $\{male = 1, child = 1\}$. This simple representation also allows for easy handling of negations, *i.e.*, "not smiling" could map to $\{smiling = -1\}$. For our initial prototype, we used a simple regular expression-based parser, which was a lookup table from various regular expressions to sets of weighted attributes.

Once we've obtained a set of attributes A_j with corresponding weights w_j, we use a search composition function C to compute the final results. The purpose of this function is to take the weighted attributes and return a score s_i for each face image I_i. We then show users the highest-scored faces. The simplest function C is the weighted product:

$$s_i = \prod_j w_j A_j(I_i) \tag{5.1}$$

The scores $A_j(I_i)$ could be taken to be the distance to the classifier decision boundary (since we use SVMs as our classifier), but this distance could mean different things for different attributes. We thus first normalize the outputs by fitting a Gaussian to the outputs of each classifier, using a held-out set of both positive and negative examples. This allows for much better rankings when multiplying several attribute values together, ensuring that the images with high confidences for *all* attributes are shown first.

At the end of this chapter, we describe some ongoing work on a more robust composition function C.

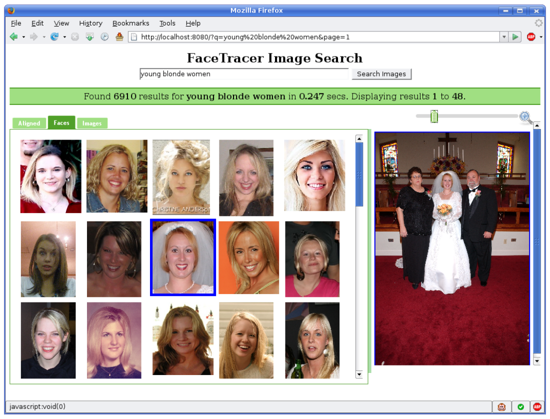
5.2 Results

Example queries on our search engine are shown in Figs. 5.1 and 5.2. The returned results are all highly relevant. Fig. 5.1d additionally demonstrates two other interesting things. First, it was run on a personalized dataset of images from a single user, showing that this method can be applied to specialized image collections as well as general ones. Incorporating our search engine into photo management tools would enable users to quickly locate sets of images and then perform bulk operations on them (e.g., edit, email, or delete). (Since current tools depend on manual annotation of images, they are significantly more time-consuming to use.) Another advantage of our attribute-based search on personal collections is that with a limited number of people, simple queries can often find images of a particular person, without requiring any form of face recognition.

Second, it shows that we can learn useful things about an image using just the appearance of the faces within it – in this case determining whether the image was taken indoors or outdoors.

5.3 Discussion

As we continue to grow and improve our system, we would also like to address some of our current limitations. First of all, any improvements in our attribute classification pipeline will, of course, directly carry-over here to improve the results. A particular aspect to highlight is the reduction of bias in the classifiers. Sometimes, due to the correlation of



http://localhost:8080/?q=young%20blonde%20women&page=1

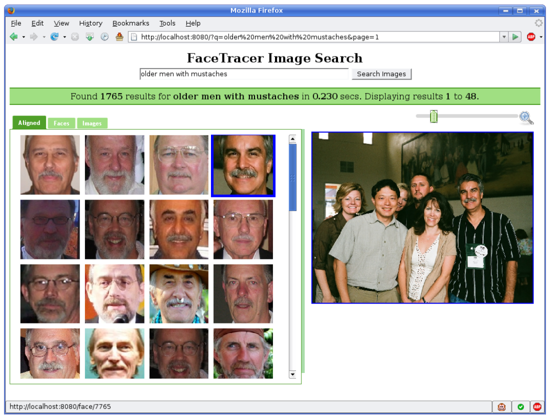
FaceTracer Image Search

young blonde women

Found 6910 results for **young blonde women** in 0.247 secs. Displaying results 1 to 48.

javascript:void(0)

(a) “young blonde women”



http://localhost:8080/?q=older%20men%20with%20mustaches&page=1

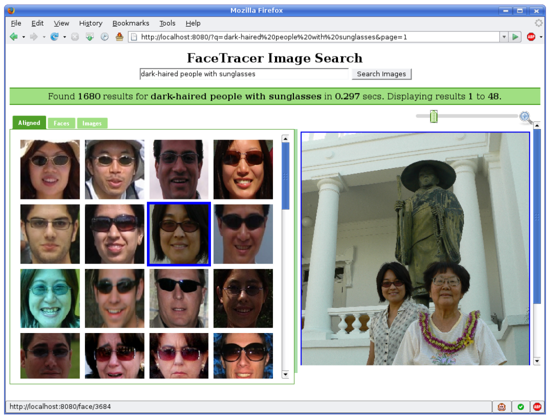
FaceTracer Image Search

older men with mustaches

Found 1765 results for **older men with mustaches** in 0.230 secs. Displaying results 1 to 48.

http://localhost:8080/face/7765

(b) “older men with mustances”



http://localhost:8080/?q=dark-haired%20people%20with%20sunglasses&page=1

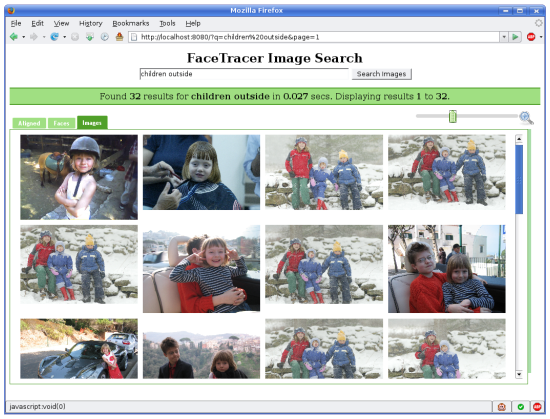
FaceTracer Image Search

dark-haired people with sunglasses

Found 1680 results for **dark-haired people with sunglasses** in 0.297 secs. Displaying results 1 to 48.

http://localhost:8080/face/3684

(c) “dark-haired people with sunglasses”



http://localhost:8080/?q=children%20outside&page=1

FaceTracer Image Search

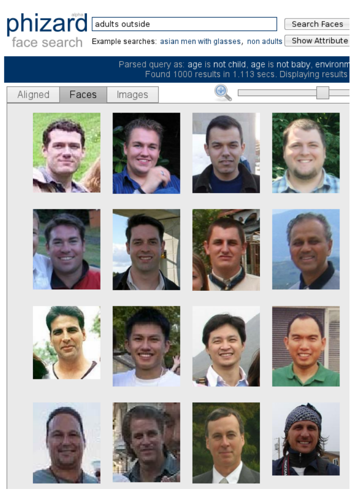
children outside

Found 32 results for **children outside** in 0.027 secs. Displaying results 1 to 32.

javascript:void(0)

(d) “children outside”

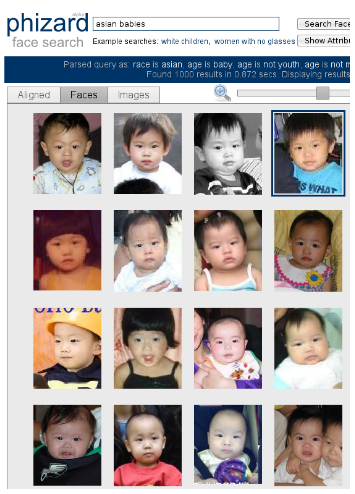
Figure 5.1: The results of various queries are shown in (a)-(c) using our attribute-based FaceTracer search engine. Note how relevant all the results are. (More results can be seen in Fig. 5.2.) The bottom-right panel (d) shows the results of the query “children outside,” applied to a single user’s personal photos. One could integrate our attribute-based search system into an image organization program such as Picasa or iPhoto to allow users access to such cabilities for their own photos. Finally, note that the results in (d) were correctly classified as being “outside” using only the cropped face images, showing that faces often contain enough information to describe properties of the image not directly related to faces.



Example searches: asian men with glasses, non adults

Parsed query as: age is not child, age is not baby, environm
Found 1000 results in 1.113 secs. Displaying results

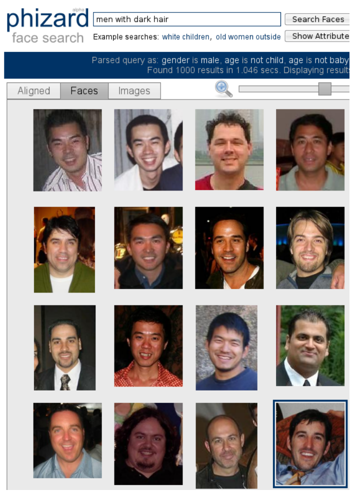
(a) “adults outside”



Example searches: white children, women with no glasses

Parsed query as: race is asian, age is baby, age is not youth, age is not m
Found 1000 results in 0.872 secs. Displaying results

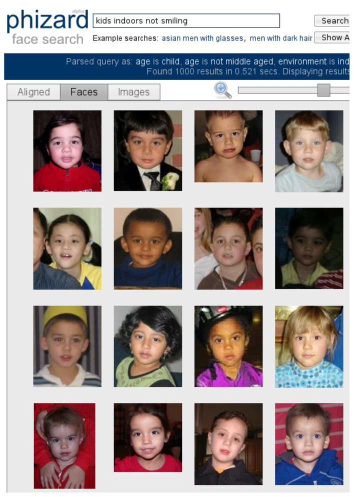
(b) “asian babies”



Example searches: white children, old women outside

Parsed query as: gender is male, age is not child, age is not baby
Found 1000 results in 1.046 secs. Displaying result

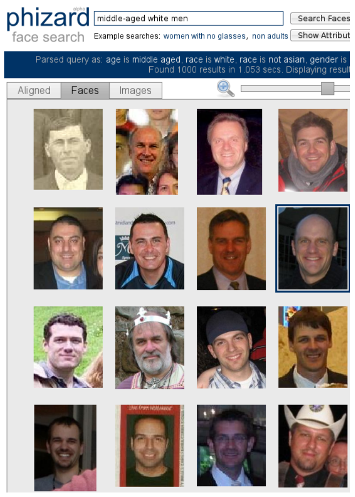
(c) “men with dark hair”



Example searches: asian men with glasses, men with dark hair

Parsed query as: age is child, age is not middle aged, environment is ind
Found 1000 results in 0.521 secs. Displaying results

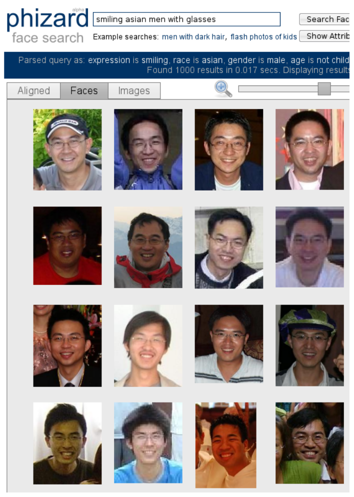
(d) “kids indoors not smiling”



Example searches: women with no glasses, non adults

Parsed query as: age is middle aged, race is white, race is not asian, gender is
Found 1000 results in 1.053 secs. Displaying resul

(e) “middle-aged white men”



Example searches: men with dark hair, flash photos of kids

Parsed query as: expression is smiling, race is asian, gender is male, age is not child
Found 1000 results in 0.017 secs. Displaying result

(f) “smiling asian men w/glasses”

Figure 5.2: More FaceTracer search results.

some attributes (*e.g.*, women tend to have thinner eyebrows), the training data can be severely biased – much more so than the inherent correlation – because labelers will pick the "easy" solution of simply labeling only males with thick eyebrows and only females with thin eyebrows. The "eyebrow thickness" classifier then becomes merely a "gender" classifier, thus throwing off results. By enforcing a good mix of genders in the training data, or being more explicit in the instructions, this bias can be reduced. See Appendix A for more details.

As far as the actual search system, both the language parser $p(Q)$ and the composition function C can be greatly improved. The former can take advantage of the many advances in natural language processing to allow for more natural and/or complex queries, for example with various conjunctions and disjunctions. It could also be extended to other languages, to facilitate non-english queries.

The second issue, the composition function, we have already begun to look at. Viewing the composition of multiple attributes as a fusion problem, we have taken advantage of recent work on robust fusion [Scheirer *et al.*, 2010] to output more reliable ranking scores. This method fits a Weibull distribution to the outputs of each classifier and then uses that to predict when individual attribute values are less reliable, resulting in a new normalization procedure that is more robust.

Another exciting direction to pursue is more fully exploring the space of faces by allowing for users to find "similar" images on the basis of some criteria – other attribute values, or low-level similarity, or some proxy for identity. If one thinks of faces contained within a hypercube where each dimension represents an attribute, then the search system we have described thus far lets users jump to any edge of this cube. But to explore inside it, we would need some other sort of mechanism, among the most natural of which would be some kind of similarity search.

This idea of similarity search plays a crucial role in the following chapter, which describes an automatic method for face replacement.

Chapter 6

Face Replacement

As a counterpoint to the search application described in the previous chapter, there are many cases where images online are *too easily available*, with huge privacy implications. Online systems such as Google Street View and EveryScape allow users to interactively navigate through panoramic images of public places created using thousands of photographs. Many of the images contain people who have not consented to be photographed, much less to have these photographs publicly viewable. Identity protection by obfuscating the face regions in the acquired photographs using blurring, pixelation, or simply covering them with black pixels is often undesirable as it diminishes the visual appeal of the image. Furthermore, many of these methods are currently applied manually, on an image-by-image basis (perhaps if someone complains). Since the number of images being captured is growing rapidly, any manual approach will soon be intractable. We believe that an attractive solution to the privacy problem is to remove the identities of people in photographs by automatically replacing their faces with ones from a collection of stock or synthetic images.

Automatic face replacement has other compelling applications as well. For example, people commonly have large personal collections of photos on their computers. These collections often contain many photos of the same person(s) taken with different expressions, and under various poses and lighting conditions. One can use such collections to create novel images by replacing faces in one image with more appealing faces of the same person from other images. For group shots, the "burst" mode available in most cameras can be used to take several images at a time. With an automatic face replacement approach, one could

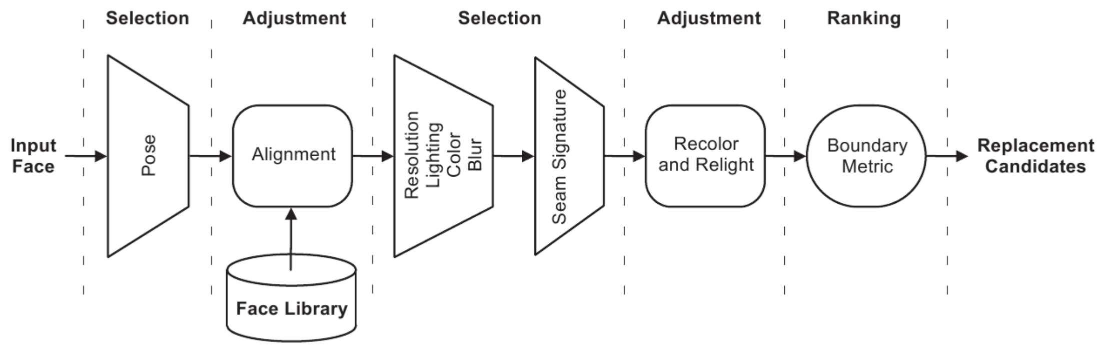


Figure 6.1: The main steps of our automatic face replacement algorithm. Given an input face that needs to be replaced, we first search the face library for faces that have similar attributes as the input face. Next, we adjust the color and lighting of the selected faces to match those of the input face. Finally, we rank the replacement candidates using a boundary metric. Given the large and diverse nature of our face library, the top ranked candidates almost always correspond to highly realistic face replacements.

create a single composite image with, for example, everyone smiling and with both eyes open.

In this chapter, we present a complete system for fully-automatic face replacement in photographs. (Figure 1.1 shows example results.) As described in Chapter 4, we have built a large library of face images which can be used for de-identification. Each face within the library is cropped from its original image, labeled with yaw and pitch pose angles estimated by the face detector, binned into one of several pose bins, and then aligned to a coordinate system common to all images in the chosen pose bin. This library can be used for our generic face replacement and de-identication applications. For the personalized replacement application mentioned previously, we can build smaller, non-generic face libraries from the users' personal photo collections.

The basic steps of our replacement approach are shown in Figure 6.1. When our system is supplied with an input image containing a face to be replaced, it performs face detection to extract the face, estimates the pose, and aligns the face to the appropriate pose bin-specific coordinate system. The system then looks into the face library to select possible candidate faces to use for replacement. Note that only candidate faces within the same pose bin of the library are considered; this ensures that replacement faces will be relatively similar in pose, thus allowing the system to use simple 2D image-compositing instead of 3D model-based approaches which require precise alignment. In addition, the system requires that the selected candidate faces are similar to the input face in image quality, color, lighting, and the boundary of the replacement region. These candidate faces can also optionally be further pruned using attribute classification outputs to ensure, *e.g.*, that gender, age, and ethnicities are preserved. Once possible candidate faces have been selected, the system transforms the color and lighting of these candidate faces to match those of the input face and blends the results into the input photograph. As a final step, to weed out inferior replacements, the system ranks the resulting images according to how well the adjusted candidate replacement face fits the surrounding region in the original photograph and chooses the highest ranked replacement.

A key contribution of our work is that it enables automatic replacement of faces across different pose, lighting, facial expression, image resolution, image blur, and skin tone – all without using 3D reconstruction techniques or manual assistance. We demonstrate how our approach can be applied to large-scale face de-identification, as well as a number of image manipulation tasks such as face swapping and creating composite group photographs. Without automation, it would be very difficult (or even impossible) to tackle these applications, setting our work in a different class from previous non-automatic approaches such as [Blanz *et al.*, 2004]. We also present results of a user study which shows that people are almost equally likely to classify real face images and our replaced face images as being real.

6.1 Appearance-Based Selection

When given an aligned input face to replace, the first step in our approach is to select candidate faces from our library which yield plausible replacements. In addition to the various attributes already mentioned in previous chapters, we define a number of new attributes that allow for more detailed measurements of the similarity of face appearances. In this section, we describe these attributes and the corresponding match criteria used to select candidate replacement faces from the library.

6.1.1 Pose, Resolution, and Image Blur

In order to produce a perceptually realistic replacement image, the poses of the input and replacement faces must be quite similar – more similar even than would be guaranteed by belonging to the same pose bin. This is because, while an in-plane rotation between the two faces can be compensated using the alignment procedure described in Chapter 4, large out-of-plane rotations, which are given by the yaw and pitch angles, are hard to adjust using image-based approaches. Therefore, we select faces from the library whose yaw and pitch angles differ by no more than $3°$ from the yaw and pitch of the original face.

It is also important to ensure that replacement faces have similar resolutions and blur properties. Significant differences in either attribute would cause a noticeable mismatch between the inside and outside regions of replaced faces. We define the resolution of facial images using the distance between the centers of the eyes. Since higher resolution images can always be downsampled, we only have to define a lower bound on the resolution of candidate faces. Therefore, we select faces from the library whose eye distance is at least 80% of the eye distance of the face to be replaced.

While there exists extensive work on estimating blur in images [Kundur and Hatzinakos, 1996; Fergus *et al.*, 2006], we use a simple heuristic metric to measure the similarity of the degree of blur in two images. This blur distance compares the histograms of the image gradient magnitude in the eye region. First, we normalize the grayscale intensity in the eye region for each of the aligned facial images to zero mean and unit variance. Second, we compute histograms $h^{(1)}$ and $h^{(2)}$ of the gradient magnitude in the normalized eye regions. Since high values of the gradient magnitude are usually associated with sharp edges, the higher-index bins of the histograms are more indicative of the blur amount. Therefore, we multiply the histograms by a weighting function which uses the square of the histogram bin index, n: $\tilde{h}^{(i)}(n) = n^2 h^{(i)}(n),\ i = 1, 2$. Finally, we compute the blur distance as the Histogram Intersection Distance (HID) [Rubner *et al.*, 2000] between the two weighted histograms, $\tilde{h}^{(1)}$ and $\tilde{h}^{(2)}$, as follows: $d_B = HID(\tilde{h}^{(1)}, \tilde{h}^{(2)})$. Only images with a weighted distance in the top 50% are kept as candidates.

6.1.2 Attributes

For many applications, it is essential to maintain certain attributes after the replacement, *e.g.*, gender, age, ethnicity. While the color and lighting adjustment steps described later can handle changes in these attributes, to some degree, the resulting images might still look unrealistic, often falling into the so-called "uncanny valley." Therefore, we can optionally filter by any of our attributes described in previous chapters.

6.1.3 Color and Lighting

The appearance of a face in a photograph is greatly affected by the incident illumination in the scene and the skin color of the face. If we attempt to replace a face with another face which was captured under significantly different illumination or with a large difference in skin color, the replacement result would appear perceptually incorrect. Although our recoloring and relighting algorithm, presented in Sec. 6.2.1, allows us to adjust for small differences in color and lighting between the two faces, drastic variations of illumination in terms of shadows and dynamic range are much harder to handle. Instead, we take advantage of the fact that our library should already contain faces captured under illuminations similar to that of the face to be replaced, and with similar skin color. For example, frontal flash images are especially common in our library, and thus our relighting technique can easily handle such cases, given that we find suitable frontal flash candidate faces. Our approach is to estimate the lighting and average color within the replacement region for each of the aligned faces in the library and, given an input face, to select faces whose lighting and color are fairly similar to the input face.

Since we only have a single image of a face, illumination in the scene cannot be accurately recovered using traditional techniques that measure or control the lighting [Debevec, 1998; Debevec *et al.*, 2000]. Instead, we use a face relighting method similar to the ones used in [Wen *et al.*, 2003] and [Wang *et al.*, 2007b]. We represent the face shape as a cylinder-like "average face shape," aligned to the coordinate system of the corresponding pose bin. We use a simple orthographic projection to define the mapping from the surface to the face. Furthermore, we assume that faces are Lambertian, and the image intensity $I_c(x, y)$ of the face replacement region in each of the RGB color channels can be approximated as $\tilde{I}_c(x, y)$

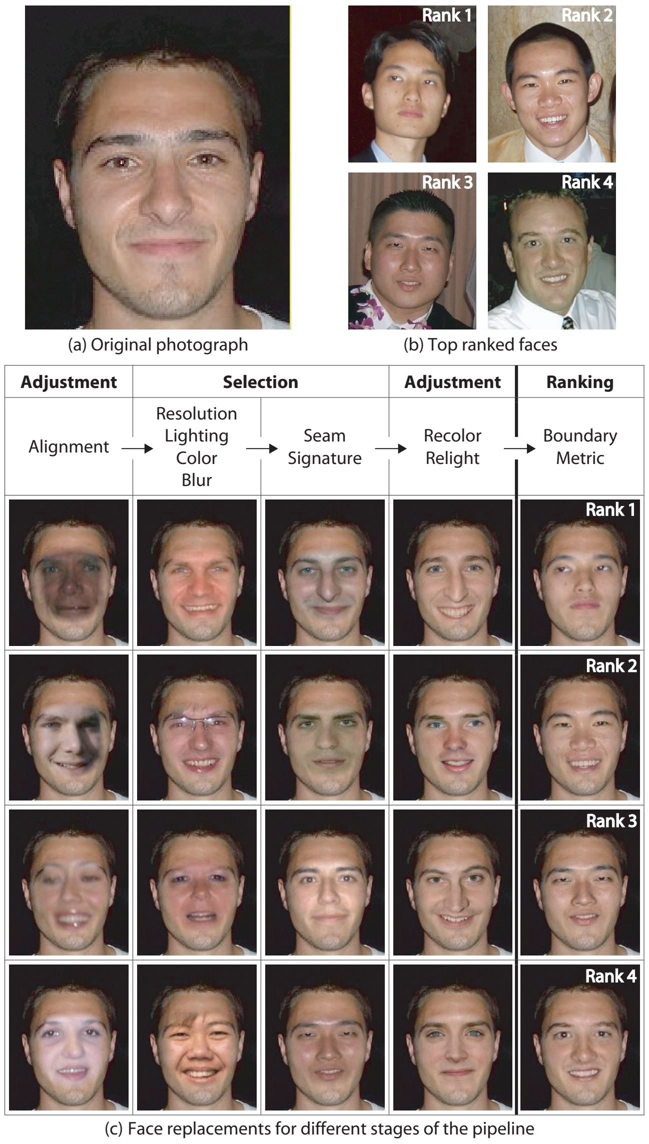


Figure 6.2: (a) An original photograph, (b) the faces for the top ranked replacements and (c) face replacement results after each step in our algorithm (as illustrated in Fig. 6.1). Each column shows the results that would be obtained if we were to not perform any of the subsequent steps (other than blending). Thus, the first column shows replacement results matching only the pose of the two faces, without any kind of selection or appearance adjustment. The subsequent columns show the results after adding basic selection, seam-signature filtering, appearance adjustments, and ranking, respectively. The results get better in each column. Note that since there is no notion of order prior to the last column, we show randomly selected replacements.

using a linear combination of 9 spherical harmonics [Ramamoorthi and Hanrahan, 2001; Basri and Jacobs, 2003]:

$$\tilde{I}_c(x,y) = \rho_c \sum_{k=1}^{9} a_{c,k} H_k(\mathbf{n}(x,y)),\ c \in \{R,G,B\}, \tag{6.1}$$

where $\mathbf{n}(x,y)$ is the surface normal at the image location (x,y), ρ_c are the constant albedos for each of the three color channels (which represent the average color within the replacement region), the coefficients $a_{c,k}$ describe the illumination conditions, and $H_k(\mathbf{n})$ are the spherical harmonic images.

Since the spherical harmonics $H_k(\mathbf{n})$ do not form an orthonormal basis in the replacement region, we cannot use the l_2 distance between the coefficients $a_{c,k}$ as a similarity measure of the lighting between two faces. Instead, we create an orthonormal basis $\psi_k(x,y)$ by applying the Gram-Schmidt orthonormalization to the harmonic basis $H_k(\mathbf{n})$. The approximate image intensity $\tilde{I}_c(x,y)$ can thus be expanded using this orthonormal basis as

$$\tilde{I}_c(x,y) = \rho_c \sum_{k=1}^{9} \beta_{c,k} \psi_k(x,y),\ c \in \{R,G,B\}. \tag{6.2}$$

We estimate the 3 albedos ρ_c and the 27 illumination coefficients $\beta_{c,k}$ by minimizing the sum of squared differences (SSD) between the right-hand side of Equation 6.2 and the aligned face image $I_c(x,y)$ within the replacement region.

We convert the RGB albedos to the HSV color space and use the l_∞ metric to compare the average color within the replacement regions of the input face image $I^{(1)}$ and the replacement candidate $I^{(2)}$. Only those candidates whose hue and saturation are within 5% and brightness within 10% of the input image are kept. To compare the illuminations, we define the lighting distance d_L as the l_2 distance between corresponding lighting coefficients (and keep only the top 50%):

$$d_L(I^{(1)}, I^{(2)}) = \left(\sum_{c \in \{R,G,B\}} \sum_{k=1}^{9} \left(\beta_{c,k}^{(1)} - \beta_{c,k}^{(2)} \right)^2 \right)^{1/2}. \tag{6.3}$$

The second column of figure 6.2c shows replacement results after selection based on resolution, blur, color, and lighting. Notice that the results are, in general, much better than those in the previous column (without pruning based on attributes).

6.1.4 Seam Signature

Although our selection process so far has already removed many unsuitable candidates for replacement, another important criteria to match is the appearance of the face along the boundary of the replacement region. Differences across this boundary (e.g., caused by facial hair, eyebrows, and hair covering the forehead) can produce visible artifacts in the final output, even after image blending. To avoid these problems, we introduce a simple filter which uses a "signature" of the seam along the replacement boundary. We first resize each aligned image in the replacement library to 256x256 pixels and then define the seam to be a strip containing all pixels inside the replacement region within a 6 pixel radius of its boundary. We create the seam signature by unfolding the seam into a rectangular image, and normalize it so that the average intensity is the same for all faces in the library. This seam signature provides an efficient representation of the texture along the replacement mask boundary. To reduce the dependence of the seam signatures on lighting, we compare the seam signatures using the L_2 distance of the absolute value of the gradient in the direction along the seam. To avoid penalizing gradual changes in appearance, we use a distance of 0 for all pixels within 8% of each other, only using the L_2 distance for pixels which differ by more than this amount. The better quality of replacement results in the third column of figure 6.2c shows that this criteria is important for filtering faces with significant differences along the boundary of the replacement region.

6.1.5 Searching the Library

Selecting candidate faces from the library using the various appearance attributes introduced in this section is a nearest neighbor search problem. This can be computationally intensive due to the high dimensionality of the blur, illumination and seam signature features. To speed things up, we use a sequential selection approach. Given a query face, we first execute a fast SQL query to select faces whose pose, resolution and average colors (given by the albedo $\rho_c, c \in \{H, S, V\}$) are close to those of the input face. This step allows us to reduce the number of potential candidate replacements from 33,000 to just a few thousand faces. Next, we further prune the list of candidates using the blur distance d_B and, subsequently, the lighting distance d_L. Finally, we select the top 50 candidate faces which match the seam signature of the input face. By running these steps in increasing order of complexity, our C++ implementation of the appearance-based selection algorithm requires less than a second to generate a list of candidate replacements for an input face image.

6.2 Appearance Adjustment and Ranking

6.2.1 Color and Lighting Adjustment

While our selection algorithm described thus far is essential for finding candidate images to replace an input face with, it is not sufficient for creating realistic results – we must adjust the lighting and color properties of the candidates to match those of the input image. This is true even with very large face libraries because the chance of finding another image with *exactly* the same lighting (with matching pose and other attributes) is extremely small. For applications with smaller libraries, such as personalized replacement, the problem is even more acute.

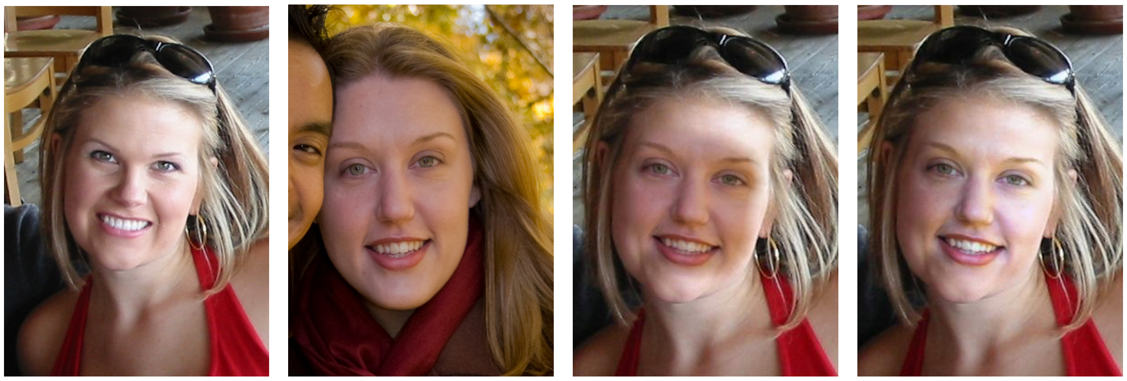
The first step in the adjustment is to use the quotient image formulation [Liu *et al.*, 2001; Wen *et al.*, 2003] to apply the lighting of the input image $I^{(1)}$ to the replacement candidate image $I^{(2)}$, within the replacement region. Using Equation 6.2, we can write the approximate image intensities for each of these images as

$$\tilde{I}_c^{(1,2)}(x,y) = \rho_c^{(1,2)} \sum_{k=1}^{9} \beta_{c,k}^{(1,2)} \psi_k(x,y),\ c \in \{R,G,B\}. \tag{6.4}$$

To obtain our relit replacement $\hat{I}^{(2)}$, we simply multiply the replacement candidate image by the ratio of the approximate images:

$$\hat{I}_c^{(2)} = I_c^{(2)} \left(\frac{\tilde{I}_c^{(1)}}{\tilde{I}_c^{(2)}} \right),\ c \in \{R,G,B\}. \tag{6.5}$$

Note that since $\tilde{I}_c^{(1)}$ and $\tilde{I}_c^{(2)}$ each capture only low-frequency lighting and color information, their ratio varies smoothly. Thus, the high-frequency content (e.g., highlights) of the replacement image is preserved during this relighting process. Furthermore, our color and lighting selection step (described in section 6.1.3) insures that the two face images are sufficiently similar in appearance that this ratio does not cause severe overflows or underflows



(a) Input (b) Candidate (c) Without adjustment (d) After recoloring

Figure 6.3: Color and lighting adjustment. We replace (a) the face in the input photograph with (b) the candidate face selected from the library. Replacement results are shown in (c) without and (d) with recoloring and relighting. Notice the significantly improved realism in the final result.

in the output. Finally, to match appearances even more closely, we transform the relit candidate image so that its RGB histogram matches that of the original input image within the replacement region.

Figure 6.3 shows the importance of our adjustment algorithm. Using the input image in Fig. 6.3a and the replacement candidate in Fig. 6.3b, we would obtain the result shown in Fig. 6.3c if we simply blended in the candidate face without performing appearance transformation. Note that even though the faces in this example have somewhat similar skin colors, the replacement result looks noticeably incorrect without adjustment. In contrast, the final replacement with adjustment, shown in Fig. 6.3d, looks highly realistic.

6.2.2 Replacement

The replacement step is simple. Since the input and candidate faces are aligned, we copy over the pixels in the corresponding replacement regions, outlined in blue in figure 3.2, from the candidate face to the input face. To ensure that the result is seamless, we apply feathering over a small number of pixels along the region boundary. Since this replacement result is in the coordinate system used for alignment, it is then transformed back to the coordinate system of the original face image. Examples of replacements created after just this alignment step – without selection and adjustment – are shown in the first column of figure 6.2c.

6.2.3 Boundary Ranking

To pick the best replacement results from our list of candidates, we rank the candidate replacements using a metric measuring their perceptual similarity to the input image along the boundary of the replacement region. This metric measures the difference between the original input and the candidate replacement to find mismatches in alignment of facial appearance (such as eyebrows), as well as occlusions and drastic lighting changes. The width of this strip is equal to 11% of the distance between the eyes (located inward from the region boundary), and this same strip is used for the final feathering operation. The ranking is computed using L_2 distance in CIE LAB space between the candidate replacement strip and the input strip. The last column of figure 6.2c shows the highest ranked results for the input face shown in figure 6.2a. The original candidate faces are shown in figure 6.2b. One can see that these results look better than the ones shown in the previous column (without ranking).

6.3 Results

Figure 6.4 shows several examples of the results obtained using our system. Each example shows, in order from left to right, the input face image, a candidate face, and the replacement result. Note the realism of the results, despite differences in pose, lighting and facial appearance. Figures 6.4c and d show examples of face replacement across different ages and genders, just to show how well the compositing works. Note that for most real-world applications, we would enforce consistency in these attributes and prevent such candidates from being used for replacement.

We now describe several applications of our system.

6.3.1 Face De-Identification

To preserve privacy in online collections of photos, one can use our system to automatically replace each face in an input image with the top-ranked candidate taken from a collection

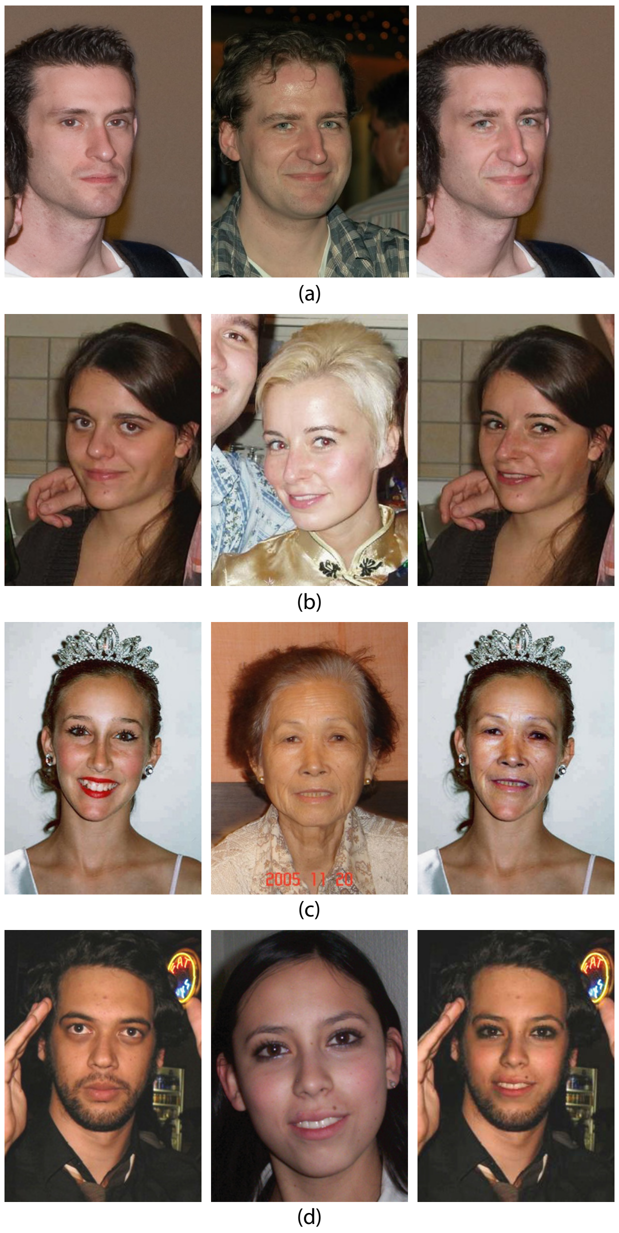
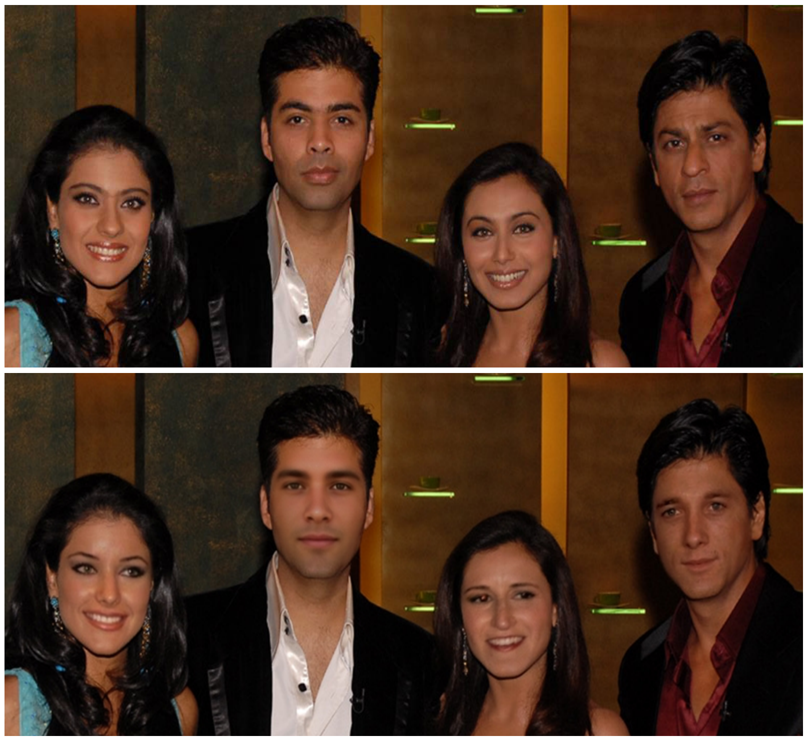
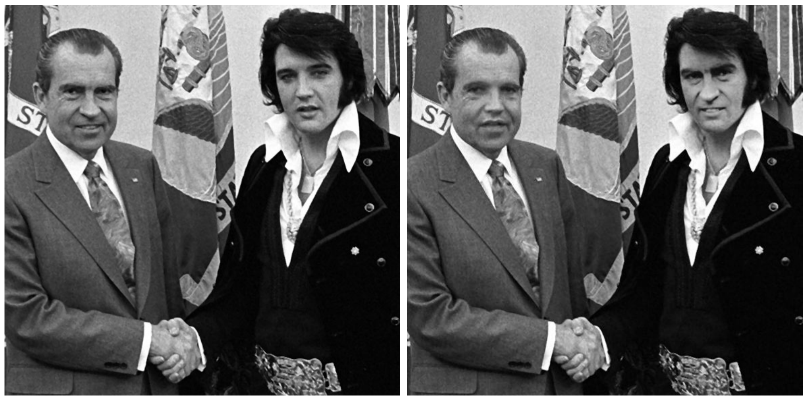


Figure 6.4: Face replacement results. Each row contains (from left to right) the original photograph, a candidate face selected from the library, and the replacement result produced automatically using our algorithm. The age and gender mismatches in (c) and (d) could be avoided by enforcing consistency across those attributes (which we disabled for these results).



(a)



(b)

Figure 6.5: Face De-Identification and Switching. (a) Result of automatically replacing the input faces (top) with the top-ranked candidate from the face library to obtain the de-identified results (bottom). No user intervention was used to produce this result. (b) Result of switching the two input faces (left) with each other to obtain the de-identified output (right). Note that it is difficult to recognize either face in the switched result.

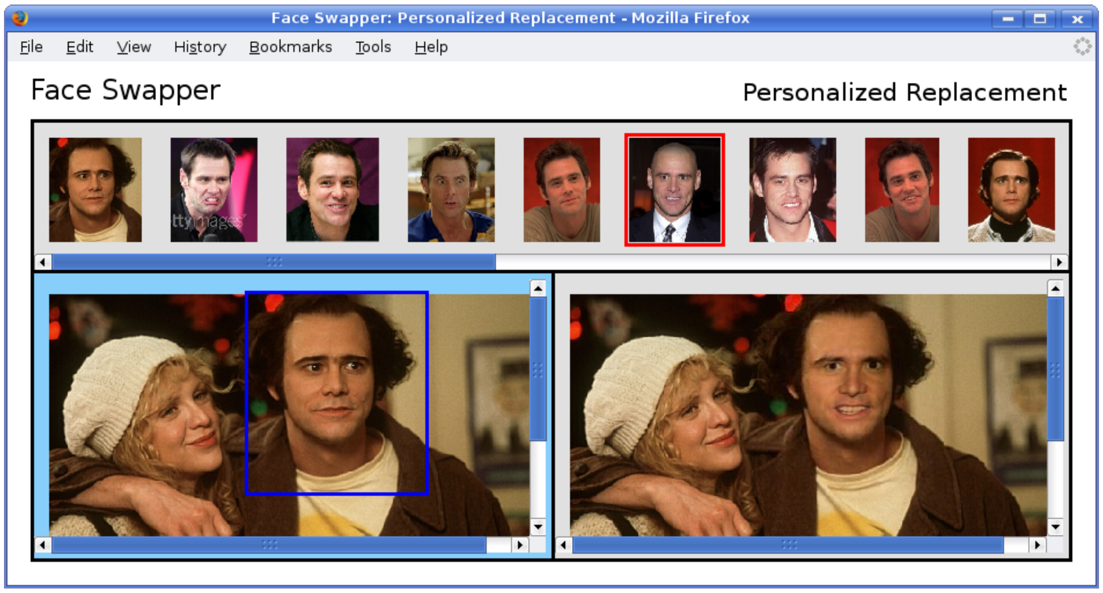

Face Swapper: Personalized Replacement - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Face Swapper
Personalized Replacement

Figure 6.6: Personalized Replacement. The face of Jim Carrey (from *Man on the Moon*, courtesy of Universal Pictures, ©1999 Universal Pictures) in the input image (left panel) is replaced using the candidate outlined in red (top panel) to create the replacement result (right panel). The list of candidates is automatically generated to match the input face.

of stock photographs. Figure 6.5a shows an example of such a replacement. We stress the fact that no user interaction was required to produce this result.

6.3.2 Switching Faces

As a special case of face de-identification (or for use as a special effect), we can limit the system to use candidates only within the same image, resulting in the switching of faces. Figure 6.5b shows the result of switching Elvis Presley and Richard Nixon's faces. Notice that neither face in the result can be readily identified as either Elvis or Nixon. (Here, we first flipped each face before replacement, so that the poses matched better.)

6.3.3 Personalized Face Replacement

With a personalized library of images, one can use our system to create novel images by replacing faces in one photograph with a more appealing one from another photo. This library can be created in a variety of ways, e.g., by using face recognition, keyword-based

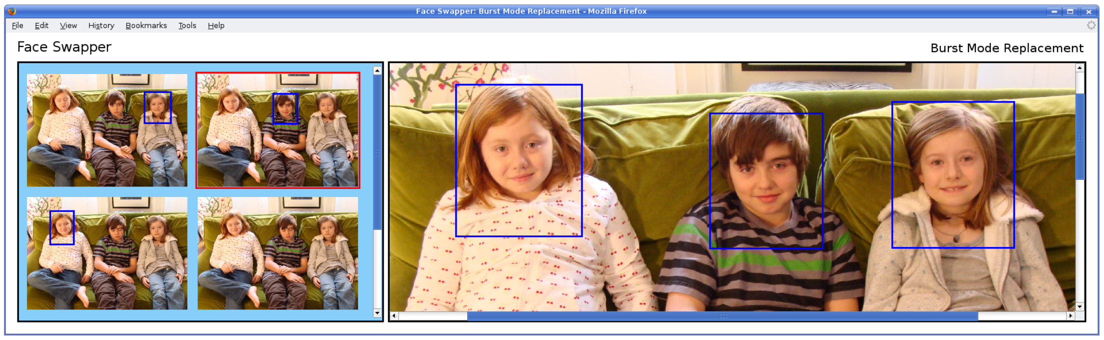

Face Swapper: Burst Mode Replacement - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Face Swapper
Burst Mode Replacement

Figure 6.7: Burst Mode Replacement. From a set of images taken using the "burst" mode of a camera (left panel), a composite image is created in which everyone is smiling and has their eyes open (right panel). The candidate faces for each child are constrained by the relative positions of the faces in all images, and thus no face recognition is required. While in this case the best replacement face for each child was selected manually (outlined in blue), blink and smile detection could be applied to select them automatically.

search, or manual labeling. figure 6.6 shows our web-based application for personalized replacement, in which we have replaced Jim Carrey's face in the original photograph (left panel) using the candidate highlighted in red (top panel). As the user clicks on the face (right panel), the replacement result for each successive candidate is shown. The list of candidates can be sorted automatically using the boundary ranking described in Section 6.2.3 to allow the user to quickly see the most compatible matches.

6.3.4 Composite Group Photographs

When taking group photographs, it is often difficult to get a "perfect" picture – where, for example, everyone is smiling, with eyes open, and looking at the camera. By taking several photos using the "burst" mode of a camera, we can construct a composite image in which everyone has the desired appearance. Since the relative position of faces does not change significantly during a burst mode shot, we limit the candidates to those with similar position in all images (thus avoiding the need for face recognition). By using the outputs of "smiling" and "blinking" attribute classifiers, we can automatically select the best face to use for each person, resulting in single-click creation of the final image. Figure 6.7 shows our implementation of this application for creating a pleasing composite image of three children

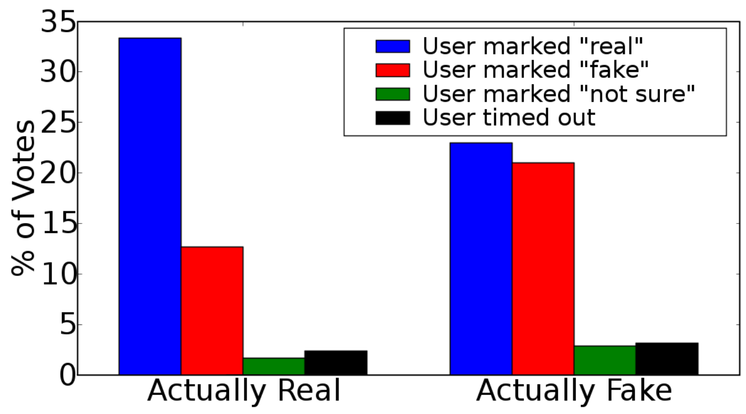


Figure 6.8: Results of the User Study. Users were presented with 50 images, half real and half fake. Users were asked to mark each image as "real," "fake," or "not sure." The first set of bars shows how the users labeled images that were actually real. The second set shows how users labeled images created by our system. Notice that users marked fake images as real almost as frequently as they marked real ones as real (58% vs. 75%). See text for details.

from a set of six input images. We see that the result, unlike each of the input images, has all of the children smiling and with eyes open.

6.3.5 User Study

The evaluation of face replacement results has so far been qualitative. To obtain quantitative results, we performed a formal user study, testing people's ability to distinguish between real images of faces and those generated by our system. For this evaluation, we showed users 50 images containing faces and asked them to classify them as "real," "fake," or "not sure," within a time limit of 5 seconds (roughly the time one would normally spend looking at a face). Exactly half of the images were real.

Across a total of 12 people tested, we found that 58% of our replaced face images were misidentified as real. In comparison, only 75% of real images were correctly marked real (full results are presented in figure 6.8). These percentages were computed as $1 - \frac{\#\ marked\ fake}{total\ number\ of\ images/2}$. Note that a vote of "not sure" or no vote within the time limit was counted as real because it suggests that a user could not definitively mark an image as fake. (Forcing users to make a decision regarding the authenticity of the image raises their sensitivity to minor appearance effects they would not normally notice.) These numbers show the high quality of our results – users could not easily differentiate between real face images and those created by our system, and often incorrectly marked real images as fake.

6.4 Discussion

We have created a comprehensive system for automatically replacing faces in photographs. Given an input image, we extract and align faces to a common coordinate system, search for similar faces from a library of candidates, match the appearance of the candidate faces using a photometric adjustment, blend the results into the input image, and finally rank the results. This entire process takes about 1 second using our C++ implementation. We note that our system is better suited for applications where the replacement face used is not specified by the user, but is allowed to be chosen (or at least narrowed) from a set of candidates. This is indeed the case with our target applications.

While we achieve very realistic results for a large variety of images, there are several limitations to our current system. These can be divided into two types: Those due to the face detector that our system depends upon, and those due to the replacement algorithm itself. Missed face detections, incorrect pose information, or misaligned fiducial point locations are all examples of problems caused by the face detector. Since the rest of the system depends on the accuracy of the detector outputs, we obtain worse results if there are any errors in them.

Figure 6.9 shows several examples of limitations of our algorithm itself. In each case, we show the input and candidate faces on top, the replacement result in the middle, and a detailed inset highlighting the problem area on the bottom. figure 6.9a shows that differences in face appearance, such as due to eyeglasses, can cause visual artifacts in replacement. Of course, by using the attribute outputs, we can easily prevent such errors. More difficult to fix is the kind of error shown in Figure 6.9b, which shows our sensitivity to occlusions, where no faces could be found in the library with a similar occlusion. Figure 6.9c shows problems as we try to replace faces in extreme poses – the face starts getting blended into the background. Finally, figure 6.9d shows a failure case for our relighting algorithm, where we forced a replacement between two faces with very different lighting. Our lighting selection step (bypassed here) would normally prevent such replacements.

In future work, we hope to remedy these issues in various ways. First, face detectors are expected to improve with time and our system will naturally benefit from this. On the algorithm side of things, our biggest limitation currently is the use of statically defined masks for each pose bin. Using dynamically-chosen optimal masks (e.g., as in [Avidan and Shamir, 2007; Efros and Freeman, 2001]) would help tremendously. These masks could also be defined hierarchically, to perform replacement on only parts of the face, thus avoiding problems with occlusions and extreme pose. Finally, using a more advanced compositing algorithm, such as Poisson blending [Pérez *et al.*, 2003], could result in more realistic outputs.

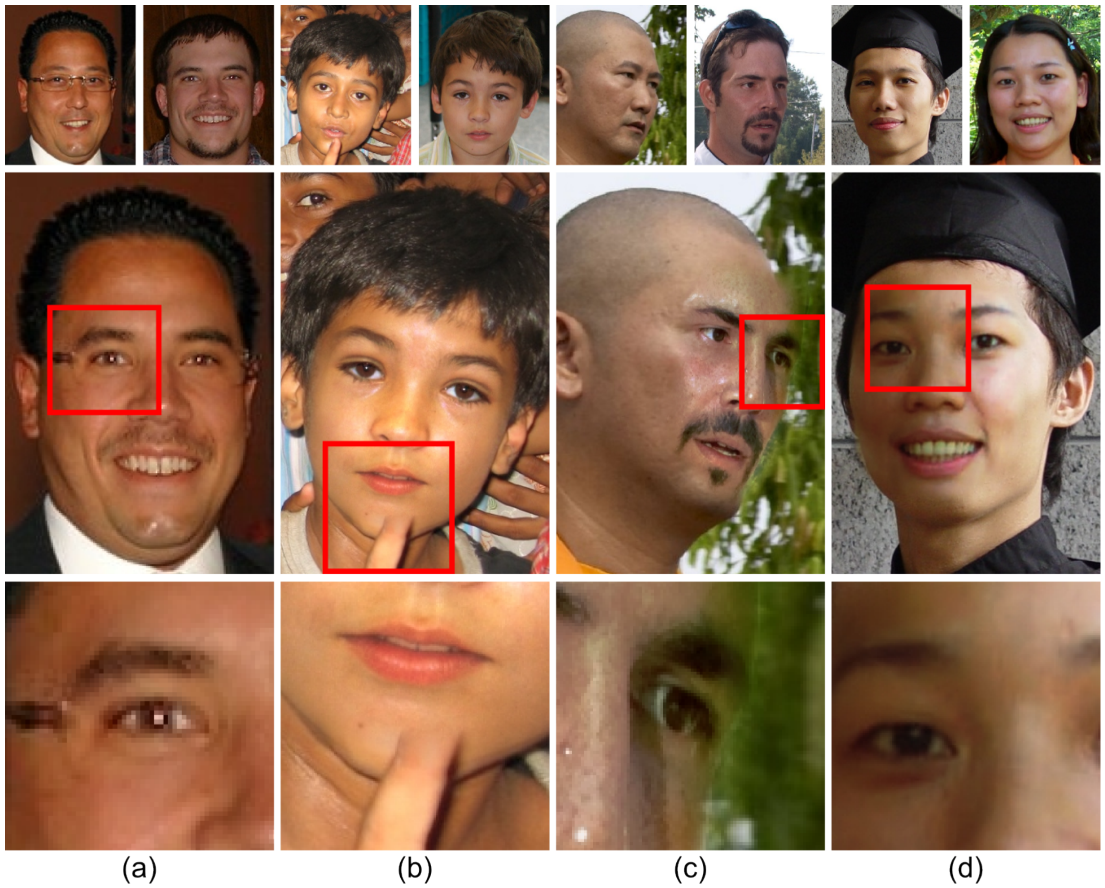


Figure 6.9: Limitations. Input and replacement candidate faces are shown on the top, replacement results in the middle, and a detailed inset of the problem area on the bottom. The lack of eyeglasses in (a) and the occluding finger in (b) cause visual artifacts in the results. In (c), the extreme pose of the face results in it being blended into the background. These problems could be solved by dynamically selecting optimal replacement regions. (d) shows a relighting failure case, caused by forcing a replacement between images with very different lighting (skipping our lighting selection step).

Chapter 7

Face Verification

There is enormous variability in the manner in which the same face presents itself to a camera: not only might the pose differ, but so might the expression and hairstyle. Making matters worse – at least for researchers in computer vision – is that the illumination direction, camera type, focus, resolution, and image compression are all almost certain to differ as well. These manifold differences in images of the same person have confounded methods for automatic face recognition and verification, often limiting the reliability of automatic algorithms to the domain of more controlled settings with cooperative subjects [Sim *et al.*, 2002; Blanz *et al.*, 2002; Phillips *et al.*, 2006; Gross *et al.*, 2001; Phillips *et al.*, 2000; Samaria and Harter, 1994; Georghiades *et al.*, 2001].

Recently, there has been significant work [Nowak and Jurie, 2007; Wolf *et al.*, 2008; Huang *et al.*, 2007b; Huang *et al.*, 2008; Huang *et al.*, 2007c] on the face verification problem – "are these two faces of the same person?" – using the "Labeled Faces in the Wild" (LFW) data set [Huang *et al.*, 2007c]. This data set is remarkable in its variability, exhibiting all of the differences mentioned above. Not surprisingly, LFW has proven difficult for automatic face verification methods [Nowak and Jurie, 2007; Wolf *et al.*, 2008; Huang *et al.*, 2007b; Huang *et al.*, 2008; Huang *et al.*, 2007c]. When one analyzes the failure cases for some of the existing algorithms, many mistakes are found that would seem to be avoidable: men being confused for women, young people for old, asians for caucasians, *etc.* On the other hand, small changes in pose, expression, or lighting can cause two otherwise similar images of the same person to be mis-classified by an algorithm as different. Based on this observation, we hypothesized that our attribute and simile classifiers (together: *traits*) could avoid such mistakes.

7.1 Training a Verification Classifier

We want to create a verification function v such that $v(I_1, I_2)$ is positive when the face images I_1 and I_2 show the same person and negative otherwise. We define this function as a particular type of composition function C which uses the trait vectors $\mathbf{A}(I)$, constructed by concatenating the result of n different attribute and/or simile classifiers:

$$v(I_1, I_2) \equiv C\left(\mathbf{A}(I_1), \mathbf{A}(I_2)\right) \tag{7.1}$$

Let $x_i = A_i(I_1)$ and $y_i = A_i(I_2)$ be the outputs of the ith trait classifier for each face ($1 \leq i \leq n$). We would like to combine these values in such a way that our verification classifier C can make sense of the data. To build C, let us make some observations about the particular form of our classifiers:

1. Values x_i and y_i will be similar if the images are of the same individual, and different otherwise.
2. Classifier values are raw outputs of binary classifiers, where the objective function is trying to separate examples around 0. Thus, the signs of values will be important.

Ideally, we would like to create contrasting outputs when the two inputs are of the same individual vs. when they are of different individuals. From observation (1), we see that using the absolute difference $|x_i - y_i|$ will yield values close to 0 when the two faces are of the same individual, and large values otherwise. From observation (2), we see that the product $x_i y_i$ will be a useful quantity, as it will be positive when both inputs have the same sign, and negative when they differ. Putting both terms together yields the tuple t_i:

$$t_i = \langle |x_i - y_i|, x_i y_i \rangle \tag{7.2}$$

The concatenation of these tuples for all n attribute/simile classifier outputs forms the input to the verification classifier C:

$$v(I_1, I_2) \equiv C(\langle t_1, \ldots, t_n \rangle) \tag{7.3}$$

$$\equiv C(|x_1 - y_1|, x_1 y_1, \ldots, |x_n - y_n|, x_n, y_n) \tag{7.4}$$

Training C requires pairs of positive examples (two images of the same person) and negative examples (images of two different people). For the classification function, we use an SVM with an RBF kernel for C, trained using libsvm [Chang and Lin, 2001] with the default parameters of $C = 1$ and $\gamma = 1/ndims$, where $ndims$ is the dimensionality of $\langle p_1, \ldots, p_n \rangle$.

7.2 Experiments

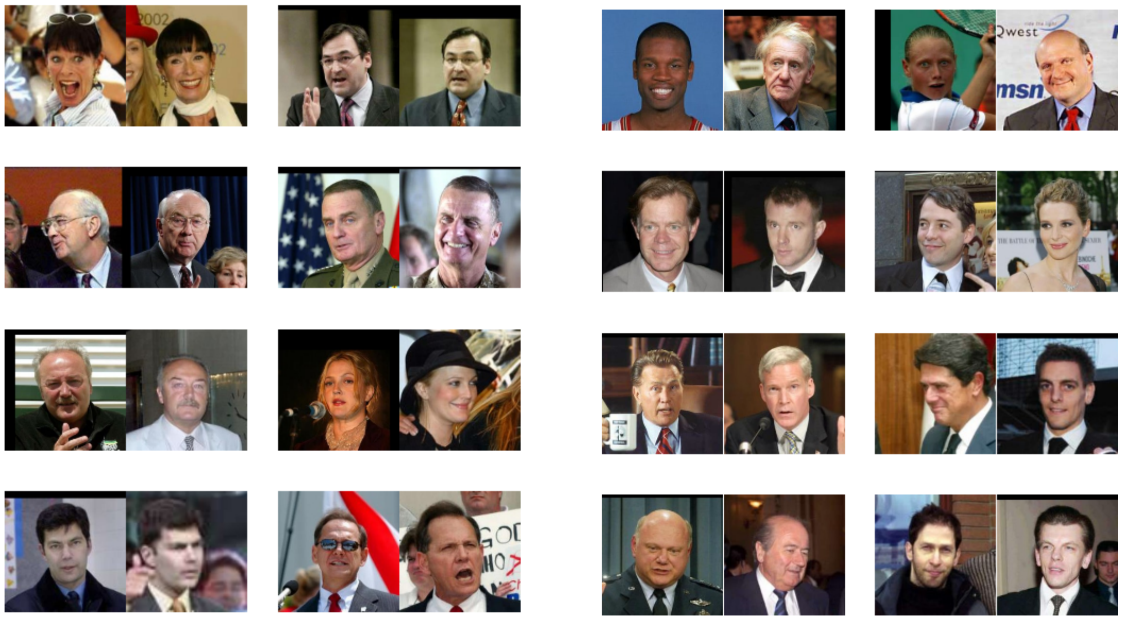
We perform face verification experiments on the Labeled Faces in the Wild (LFW) benchmark [Huang *et al.*, 2007c] and also on our PubFig benchmark. For each computational experiment, a set of pairs of face images is presented for training, and a second set of pairs is presented for testing. In all experiments, not only are the images in the training and test sets disjoint, but there is also no overlap in the individuals used in the two sets. In addition, the individuals and images used to train the attribute and simile classifiers are disjoint from the testing sets.

7.2.1 Attribute Classifier Results on LFW

The LFW dataset consists of 13,233 images of 5,749 people, gathered from news photos, and organized into 2 "views":

1. A development set of 2,200 pairs for training and 1,000 pairs for testing, on which to build models and choose features; and
2. A 10-fold cross-validation set of 6,000 pairs, on which to evaluate final performance.

We used View 1 for high-level model selection (*e.g.*, representation for the final classifier C) and evaluated our performance on each of the folds in View 2 using the "image



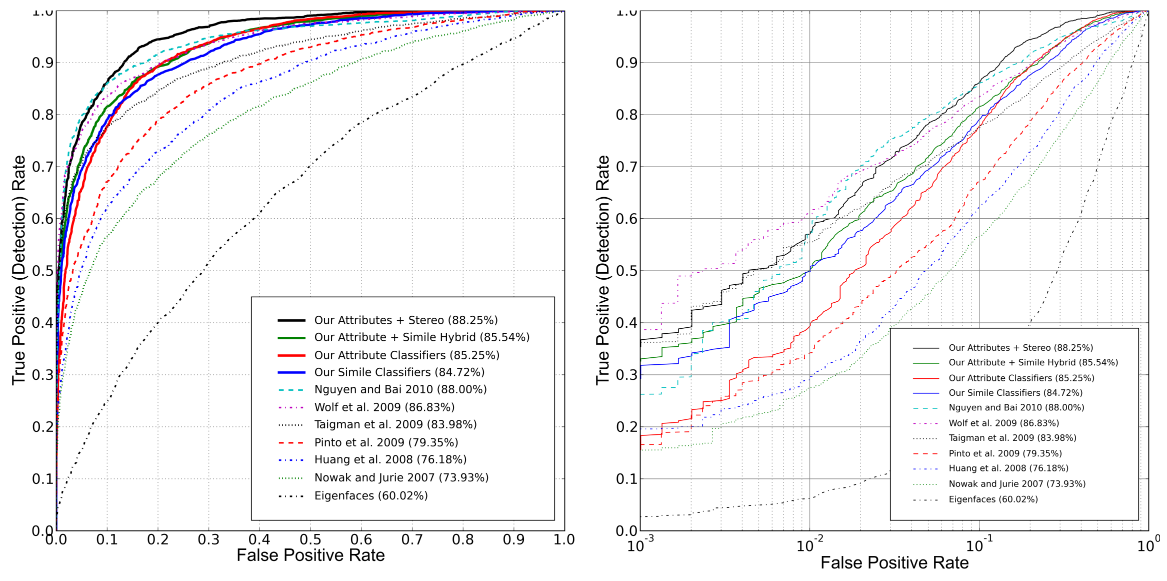
(a) “Same” pairs (b) “Different” pairs

Figure 7.1: Randomly chosen (a) “same” and (b) “different” pairs from the Labeled Faces in the Wild (LFW) benchmark [Huang *et al.*, 2007c]. Note the difficulty of this dataset, with extensive variation in view, illumination, expression, image quality, age, *etc.*

restricted configureation,” as described in the LFW paper [Huang *et al.*, 2007c]. Some sample verification pairs from LFW are shown in Fig. 7.1.

A verification classifier C is trained using nine folds from View 2 of LFW and then evaluated on the remaining fold, cycling through all ten folds. Receiver Operating Characteristic (ROC) curves are obtained by saving the classifier outputs for each test pair in all ten folds and then sliding a threshold over all output values to obtain different false positive/detection rates. An overall accuracy is obtained by using only the signs of the outputs (*e.g.*, thresholding at 0) and counting the number of errors in classification. The standard error is computed as described in the LFW paper [Huang *et al.*, 2007c].

Fig. 7.2 shows results on LFW for our attribute classifiers (red line), simile classifiers (blue line), a hybrid of the two (green line), and attributes + stereo (black line) along with several previous methods (dotted lines) [Nguyen and Bai, 2010; Wolf *et al.*, 2009; Taigman *et al.*, 2009; Wolf *et al.*, 2008; Pinto *et al.*, 2009; Huang *et al.*, 2008; Nowak and Jurie, 2007; Turk and Pentland, 1991]. The accuracies for each of our methods are $85.25\% \pm 0.60\%$,



(a) LFW results, linear scale (b) LFW results, log scale

Figure 7.2: Face verification performance on LFW of our attribute classifiers, simile classifiers, a hybrid of the two, and attributes + stereo are shown in solid red, blue, green, and black, respectively, on (a) linear and (b) log-scales. Dashed lines are existing methods. Our highest accuracy is 88.25%, which is beats the current state-of-the-art accuracy of 88.00% [Nguyen and Bai, 2010]. Notice that similes perform better at low false positive rates, attributes better at high detection rates, and the hybrid and stereo combinations better throughout.

$84.72\% \pm 0.41\%$, $85.54\% \pm 0.35\%$, and 88.25%, respectively.[1]

Our highest accuracy of 88.25% is higher than the 88.00% accuracy of the current state-of-the-art method [Nguyen and Bai, 2010] on LFW. This method uses our attributes combined with pixel-matching costs obtained using the stereo-matching method of Castillo and Jacobs [2007]. This method treats a verification pair as a stereo pair (explicitly assuming non-rigidity). It then uses standard stereo-matching methods to get costs between the faces for each scanline, and uses the sum of the costs as a measure of how similar the two faces are. This combination performs best overall because the two components (attributes and stereo matching) complement each other. When two faces match in most attributes, the attributes alone would be unable to make the fine-scale distinctions necessary to disambiguate faces,

[1]Our face detector was unable to detect one or more faces in 53 of the 6,000 total pairs. For these, we assumed average performance.

but this is where the stereo method excels. In contrast, when faces are quite different, the stereo method by itself might be unable to handle the large variations in appearance and expression and thus this regime is more suited to the attributes.

The small bump in performance from combining the attribute and simile classifiers in the hybrid method suggests that while they contain much of the same kind of information, there are still some interesting differences. This can be better seen in Fig. 7.2b, where similes do better in the low-false-positive regime, but attributes do better in the high-detection-rate regime. One way to think about why this hybrid can do better than either attributes or similes alone is via an example. For instance, it is possible for two people of different genders to have eyes like Salma Hayek's and noses like Meryl Streep's. So, while the simile classifier might confuse these, the attribute classifier would not. Conversely, two dark-haired women with big lips might have very similar attribute values, but one might look more like Angelina Jolie and the other more like Salma Hayek. In this case, the simile classifiers would be able to disambiguate the two individuals.

7.2.2 Human Attribute Labels on LFW

Although our methods already achieve close to the current best performance on LFW, it is interesting to consider how well attribute classifiers could potentially do. There are several reasons to believe that our results are only first steps towards this ultimate goal:

- We have currently trained 73 attribute classifiers. Adding more attributes, include fine-scale ones such as the presence and location of highly discriminative facial features including moles, scars, and tattoos, should improve performance [2]. Also, our classifiers are not perfect, and so there is room for improvement in their accuracies, which should carry over to verification performance as well.

- Of the 73 attributes, many are not discriminative for verification. For example, facial expression, scene illumination, and image quality are all unlikely to aid in verification.

[2]We ran a preliminary evaluation on the usefulness of these fine-scale features and found that they are not as useful on LFW due to the low quality of images. However, there are many operational scenarios where datasets will be of much higher quality, thus making these attributes more relevant in those cases.

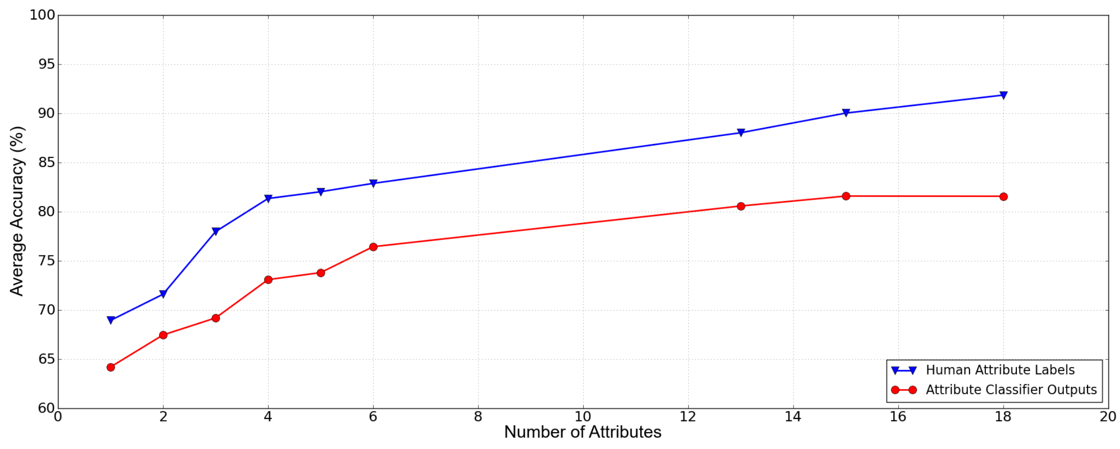


Figure 7.3: Comparison of face verification performance on LFW using human attribute labels (blue line) vs. automatically-computed classifier outputs (red line). Verification using human labels consistently outperforms that using classifier outputs. With 18 attributes, human attribute labels reach 91.86% accuracy, compared to only 81.57% using classifier outputs. Training better attribute classifiers (or regressors) could thus greatly improve verification performance.

There is also a severe imbalance in LFW of many basic attributes such as gender and age, which reduces the expected benefit of using these attributes for verification.

- The attribute functions were trained as binary classifiers rather than as continuous regressors. While we use the distance to the separation-boundary as a measure of degree of the attribute, using regression may improve results.

With the hope of exploring what might be possible given better attribute classifiers, we performed an experiment in which our automatic attribute labeling process was replaced by human labels, keeping the verification process identical. MTurk workers were asked to label attributes for all faces in the LFW View 2 benchmark set. We averaged seven user-responses per image to obtain smoothed estimates of the attribute values.

Fig. 7.3 shows a comparison of face verification performance on LFW using either these human attribute labels (blue line) or our automatically-computed classifier outputs (red line), for increasing numbers of attributes. In both cases, the labels are fed to the verification classifier C and training proceeds identically, as described earlier. The set of attributes used for each corresponding point on the graphs were chosen manually (and were identical for both). Verification results using the human attribute labels reach 91.86% accuracy with 18 attributes, significantly outperforming our computed labels at 81.57% for the same 18 attributes. Moreover, the increase in accuracies from computational to human labels is actually *increasing* with more attributes, suggesting that adding more attributes could further improve accuracies.

7.2.3 Human Verification on LFW

The high accuracies obtained in the previous section lead to a natural question: How well do people perform on the verification task itself? While many algorithms for automatic face verification have been designed and evaluated on LFW, there are no published results about how well people perform on this benchmark. To this end, we conducted several experiments on human verification.

We followed the procedure of O'Toole *et al.* [2007] to obtain this data, using Amazon Mechanical Turk. MTurk users were shown pairs of faces from the LFW View 2 benchmark set and asked to mark whether the images showed the same person or not. This was done on a scale of -1 to $+1$, where the sign of the score was their decision, and the magnitude was their confidence in their response. The responses of 10 different users were averaged per face pair to get a score for that pair. (Thus, for the $6,000$ image pairs in LFW, we gathered $60,000$ data points from users for each of the three tests described below, for a total of $240,000$ user inputs.) An ROC curve was created by sliding the confidence threshold from -1 to $+1$, counting scores less than the threshold as "different" and those above as "same."

Results are shown in Fig. 7.4. Using the original LFW images (red curve), people have 99.20% accuracy – essentially perfect[3]. We then made the task tougher by cropping the images, leaving only the face visible (including at least the eyes, nose and mouth, and possibly parts of the hair, ears, and neck). This experiment measures how much people are helped by the context (sports shot, interview, press conference, *etc.*), background (some images of individuals were taken with the same background), and hair (although sometimes

[3]We submitted a similar job which also asked the users if they recognized either person in each pair, and averaging the responses of only the unrecognized pairs yielded almost exactly the same performance, suggesting that this is an accurate *verification* (as opposed to *recognition* result.

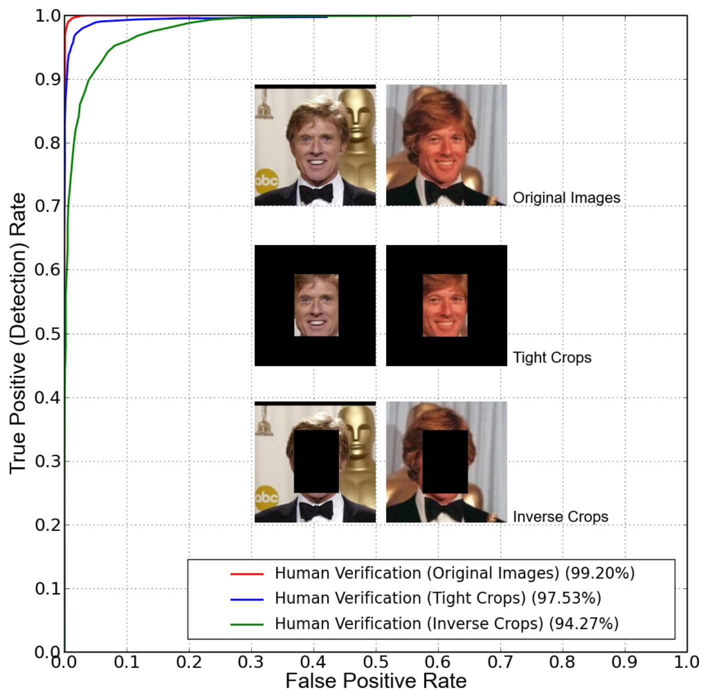


Figure 7.4: Face verification performance on LFW by humans is almost perfect (99.20%) when people are shown the original images (red line). Showing a tighter cropped version of the images (blue line) drops their accuracy to 97.53%, due to the lack of available context. The green line shows that even with an inverse crop, *i.e.*, when *only* the context is shown, humans still perform quite well, at 94.27%. This highlights the strong context cues available on the LFW dataset.

it is partially visible). The results (blue curve) show that performance drops to 97.53% – a tripling of the error rate.

To confirm that the region outside of the face is indeed helping people with identification, we ran a third experiment where the mask was inverted, *i.e.*, we blacked out the face but showed the remaining part of the image. Surprisingly, people still achieve 94.27% accuracy, as shown by the green line in Fig. 7.4. These results reinforce the results of Sinha *et al*. [Sinha *et al.*, 2006], that context and hair are powerful cues for face recognition. It also perhaps points to a bias in LFW – many news photos tend to be taken at the same event, making the face recognition task easier.

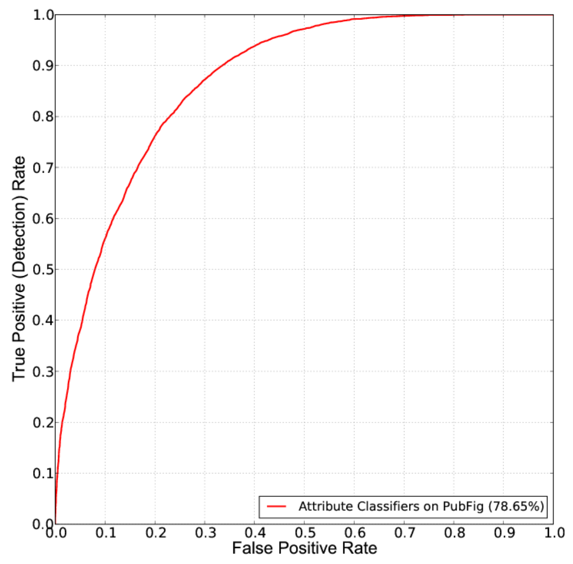


Figure 7.5: Face verification results on the PubFig evaluation benchmark using our attribute classifiers. Our accuracy is 78.65% on this benchmark, which consists of 20,000 face pairs partitioned into 10 folds for cross-validation. Our lower performance on this experiment as compared to LFW suggests that it is a more challenging dataset.

7.2.4 Attribute Classifier Results on PubFig

The PubFig dataset, being much deeper (more images per person) and gathered from more varied sources, should ameliorate this issue. We test this hypothesis using an evaluation benchmark similar to LFW's. Face verification is performed on 20,000 pairs of images of 140 people, divided into 10 cross-validation folds with mutually disjoint sets of 14 people each. These people are separate from the 60 people in the development set of PubFig, which were used for training the simile classifiers. The performance of our attribute classifiers on this benchmark is shown in Fig. 7.5, and it is indeed much lower than on LFW, with an accuracy of 78.65%.

7.3 Discussion

We have presented and evaluated two approaches for face verification using traits computed on face images – based on describable attributes and our novel simile classifiers. This is the first time such attributes have been applied to face verification. Both approaches result in error rates significantly lower (14.75% to 14.46%) than the state-of-the-art for face verification on the LFW data set. Furthermore, this is achieved using only the face region of images (without the background or context). This is important because our experiments measuring human performance show that people perform suprisingly well (94.27%) at this task even if the central portion of the face is artificially occluded. However, humans perform quite well (97.53%) when shown only a tight crop of the face, leaving a great deal of room for improvement in the performance of algorithms for face verification in the unconstrained setting.

One somewhat surprising result was that using all attributes almost always outperforms any subset of them – even though many of the attributes should not be related to the identity of the person. There are several reasons for this. One is that machine learning algorithms in general perform better given more input features (*e.g.*, overfitting). But more encouragingly, our intuition is that many non-identity-related attributes are still useful for making a final verification decision. For example, knowing that one image is blurry while the other is not lets the classifier perhaps put less emphasis on finer-scale attributes that might not be as reliably detected (this is possible due to our use of RBF-kernel classifiers; linear classifiers might not be able to take advantage of such combinations). Finally, another possibility is that *within a particular dataset*, some of these attributes are actually discriminative. For example, it could be that certain individuals are always photographed smiling, or outdoors, or with mouth open, and so these attributes are useful for identification.

Another interesting result (not described above) is an experiment where we thresholded the attribute outputs and measured verification performance. We wanted to see how much the exact value of the attribute classifier outputs matters to verification performance, as opposed to just a signed boolean value. This resulted in an accuracy drop of about 5%, suggesting that a true binary classifier (*i.e.*, without using the distance-to-the-margin trick that we do) would not be as effective.

Despite our best efforts, our estimators will not perfectly measure all attributes all the time, especially in challenging imaging conditions. Therefore, we could ask humans to look at misclassified pairs of images and suggest new attributes for discrimination. We can then train estimators for these new attributes, and learn a better verification classifier V. With this dynamic feedback loop, the system would be able to improve itself over time, targeting exactly the cases with which it has the most trouble. A similar approach was implemented in recent work for attributes in the general case [Parikh and Grauman, 2011], showing exactly this expected behavior.

The verification framework is well suited for cases where we have only a few images per class – a test image can be identified by verifying it against all training images, and picking the class of the best match. However when we have many training examples per class, using attributes also opens up the possibility of learning which attributes are important for each class. For example, some individuals consistently display extreme values for certain attributes, and one can thus place more weight on these attributes for reliable identification, *e.g.*, Woody Allen's glasses and receding hairline, Angelina Jolie's large lips and eyes, or Groucho Marx's eyebrows and mustache.

These reliably distinctive attributes suggest investigating approaches that can adapt to intra-class and inter-class variation, either by learning a global rescaling as in Fisher Linear Discriminants [Fisher, 1936], Kernelized Fisher Discriminants [Yang *et al.*, 2005], or locally as in [Frome *et al.*, 2007] or a non-linear SVM. We plan to explore all of these, especially considering methods that can adapt to the dependencies that may be present between attributes.

Finally, it is worth expanding the notion of face recognition to person recognition – *i.e.*, using cues other than just the face itself. In still images, this could include body shape, clothing (or typical clothing styles), background, context, *etc.* Of course, many of these aspects of personal appearance can change from image to image. However, there are many scenarios where they might be reasonably constant. For example, when organizing a personal photo collection, EXIF tags on images can be used to group them into sets taken in the same session.

Moving to video, many more possibilities open up. While the simplest thing to do would be to compute attributes on each frame individually and then combine them at a later stage using some sort of classifier, more sophisticated approaches could yield even better results.

Chapter 8

Plant Species Identification

We finally come full-circle back to our motivation for attributes: plant species identification. As we described in the introductory chapter, the use of attributes has been around since antiquity for this task. Unfortunately, the attribute classification process for plants is currently time-consuming, brittle, and frustrating for most people. Users must manually traverse a decision tree in a printed field guide to make an identification. This process is fraught with possible errors due to individual variations in the particular plant one is looking at, mistakes in the user's interpretation of the attributes, ambiguous descriptions of characteristics, and the lack of detail in the diagrammatic illustrations typically shown in field guides (as opposed to high-quality photographs).

To bring this process into the modern era, we have adapted our automatic attribute classification framework to build *Leafsnap*, an electronic field guide, which runs as an application on mobile devices such as the iPhone and iPad. Leafsnap currently contains high-quality photographs and descriptions of all the vascular tree species of the northeast United States (around 200 or so). This already represents a huge step-up from a traditional field guide: as a software application, operations such as browsing, sorting, and searching are trivial. With complete coverage of all aspects of each plant – the leaf, flower, fruit, petiole, bark, *etc.*– users can quickly compare a specimen with the provided photographs to see if it looks like the same species.

We can then take this application to the next level by giving users access to an automatic identification system. This is done by letting users take photos of leaves, which are then sent to our server. On the server, we segment the leaf from the background, compute low-level features on the segmented leaf, compute high-level attributes on these features, and finally perform identification, returning the top-ranked species back to the user. Of course, since this process is not perfect, the final identification must be done by the user – but the task is now much easier.

The benefits of Leafsnap to the end-user are obvious and significant: they now have available at their fingertips access to a field guide with high-quality, attractive photos, advanced search capabilities, and automatic recognition of all plant species in their area. The app also includes compelling games that help train users in learning to recognize and distinguish different plant species on the basis of their leaves, fruits, or flowers. The whole package also makes a much better introduction to botany and taxonomy for children. Existing, printed field guides, while adequate for recognition by adults and those with experience in classification, are often too difficult for children to use. Perhaps even more importantly, they are not very inviting to kids, and the diagramatic images of plants they contain are neither compelling nor life-like.

Our work also has potential impact on both professional botanists and enthusiasts. The international biodiversity community has expressed a particularly enthusiastic response to our current system. Even though the device is still in prototype form, field botanists have recognized its potential for addressing the current taxonomic impediment to fast and accurate species identification. For example, at a recent professional meeting of tropical biologists in Mexico, our collaborators demonstrated our system at a poster session with hundreds of presentations and were overwhelmed with requests to build similar systems for field sites in Brazil, Costa Rica, Mexico, and Panama.

Given a new image, the recognition process consists of:

1. Classifying whether the image is of a leaf, to decide if it is worth processing further. This is done using a leaf/non-leaf attribute classifier which uses the Gist feature [Oliva and Torralba, 2001] on the image.

2. Segmenting the image to obtain a binary image separating the leaf from the background. This is currently implemented using an Expectation-Maximization frame-

work, estimating foreground and background color distributions in the HSV colorspace.

3. Extracting features from the binarized image for compactly and discriminatively representing the shape of the leaf. We use histograms of curvature over scale as the feature representation, robustly and efficiently implemented using integral measures of curvature.

4. Optionally, computing attributes on either these low-level features, or the input image directly to aid in recognition.

5. Comparing the features and attributes to those from a labeled database of leaf images and returning the species with the closest matches. Due to the discriminative power of the features and the size of our labeled dataset, we currently use a simple nearest neighbor approach with the L_1-norm.

This whole process is completed in under 5 seconds, and can be trivially parallelized across many machines.

8.1 Curvature Histograms over Scale

Curvature is a fundamental property of shape and has thus attracted much attention from the vision community. However, when dealing with images on discrete pixel grids, the elegant mathematical definitions of curvature are tricky to implement in a stable manner. Typically, curvatures are computed using differential techniques, which amplify noise, are sensitive to the orientation and scale of captured images, and are difficult to define at multiple scales.

Instead, we can use *integral measures* to compute functions of the curvature at a boundary point [Manay *et al.*, 2006; Pottmann *et al.*, 2009]. One such measure in 2D is the intersected portion of a disk centered at a contour point and the inside of the contour. For straight, concave, and convex boundaries, the fraction of the disk intersected will be $=$, $<$, or > 0.5, respectively. Furthermore, this representation lends itself naturally to a notion of scale: the radius of the disk. Perturbations much smaller than the disk radius are ignored.

Thus, a serrated boundary will show large, alternating curvatures at a fine scale, but a smooth line at a coarse scale.

There are many advantages to using such integral measures instead of their differential counterparts: they are easy and fast to compute for images on discrete grids; small discretization errors and noise disappear at larger scales; complex topology poses no problem; and there exist straightforward analogues in 3D [Pottmann *et al.*, 2009].

Given this robust method for computing curvatures, what representation should we use for our low-level features? One simple representation we could use is the concatenation of curvature values along the contour of the object, into a single feature vector. But several things can go wrong with this approach: different contours will have different lengths, making them difficult to compare; contours must be aligned to have the same starting point, otherwise they will not match; minor changes in topology or orientation can cause huge changes in the feature vector; and there is no straightforward way to handle multiple contours.

Instead, we compute histograms of the curvature values at each scale, and concatenate these histograms together to form the Histogram of Curvatures over Scale (HoCS) feature. Histograms have the benefit of being compact, simple to represent, not requiring alignment, fast to compare using metrics such as L_1, and being widely used for a variety of tasks in computer vision and other areas.

The benefits of the HoCS feature are demonstrated in Fig. 8.1, which shows segmented images of four different leaf species, and curvature histograms for each shape computed at two different scales – coarse (large radius) on top, and fine (small radius) on bottom. The pair of images in each row share the same general shape, but the images on the left have a smooth boundary, and the ones on the right have a serrated boundary. Thus, the histograms of coarse-scale values for each *row* are similar, while for each *column*, the histograms of fine-scale values are similar. By using both histograms together, we can distinguish each of these shapes from each other. (Our actual HoCS features use several more scales – 31.)

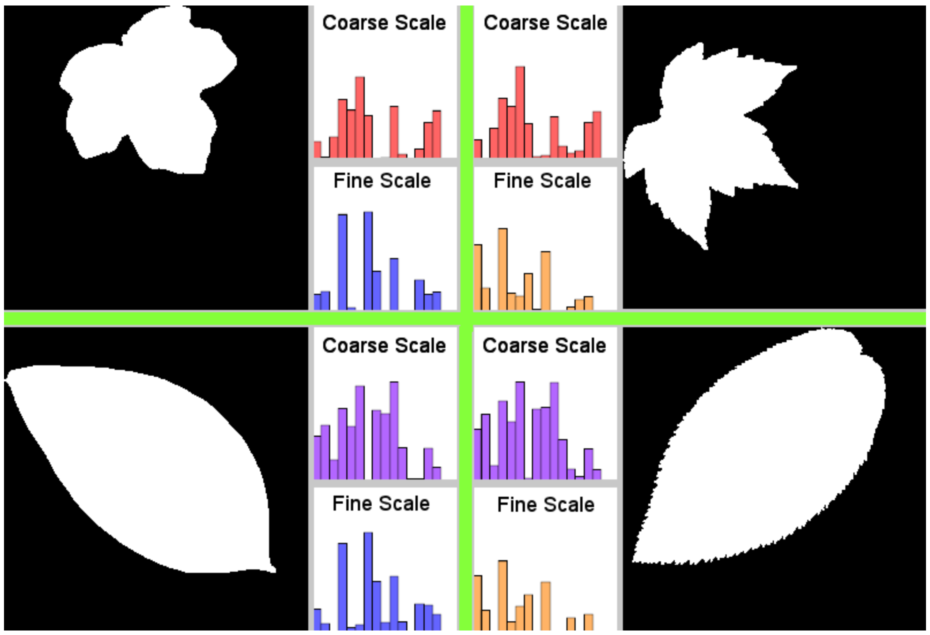


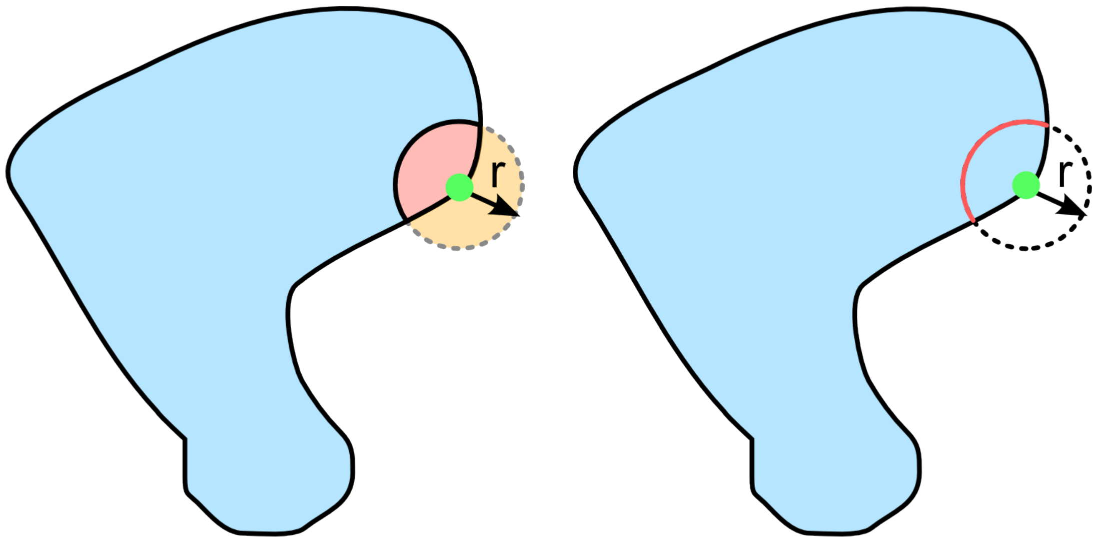
Figure 8.1: For these leaves of four different species, we show histograms of curvature at two different scales (coarse on top, fine on bottom). Notice that in each *row*, the two leaves share the same coarse shape (5-lobed or 2-lobed) and hence the top histograms match. Similarly, in each *column*, the fine-scale curvatures are similar (smooth or serrated) and hence the bottom histograms match. Taking both histograms together, none of the leaves match each other, despite their similar appearance at one scale. We adapt this idea for leaf identification by robustly computing histograms of curvature over many scales as our low-level features.

8.1.1 Computing Curvatures

All integral measures are computed by placing a disk of radius r at the point where we wish to measure curvature. The different types of measures are:

1. **2D Area Measure:** The fraction of the disk's area inside the contour (see Fig. 8.2a)
2. **2D Arclength Measure:** The fraction of the disk's perimeter inside the contour (see Fig. 8.2b)

While these definitions are simple to state, there are several issues one must keep in mind in order to get accurate measurements. For scale invariance in 2D, we first resize all images to a common area before extracting curvatures. To ensure consistency in results, we use fixed-size disks (spheres) to perform calculations. To remove problems caused by "holes"



(a) Area Measure (b) Arclength Measure

Figure 8.2: (a) The 2D area measure is the fraction of the circle's area contained inside the object (shown in pink). (b) The 2D arclength measure is the fraction of the circle's perimeter contained inside the object (shown in red). In both cases, the notion of scale is determined by the radius r of the circle. 3D measures can be defined analogously.

in the shape, we completely fill-in the contours prior to resizing and curvature extraction. To prevent histogramming artifacts, we use bilinear interpolation to do soft-binning of curvature values.

The simplicity of these measures also makes several speed optimizations possible:

1. [**2D Arclength Measure**]: We only look for intersections at least r contour points away from the central point.

2. [**2D Arclength Measure**]: We precompute all disk intersection points and do $O(1)$ lookups at runtime.

3. [**2D Area Measure**]: The change in intersected area from one contour point to the next can be computed by simply checking the crescent-shaped boundaries of the circle in the direction of the shift. Since the change in position is limited to 4 or 8 different translations (depending on the connectedness of the contour points) and our set of radii are fixed, we precompute these coordinates, resulting in approximately $2\pi r$ pixels

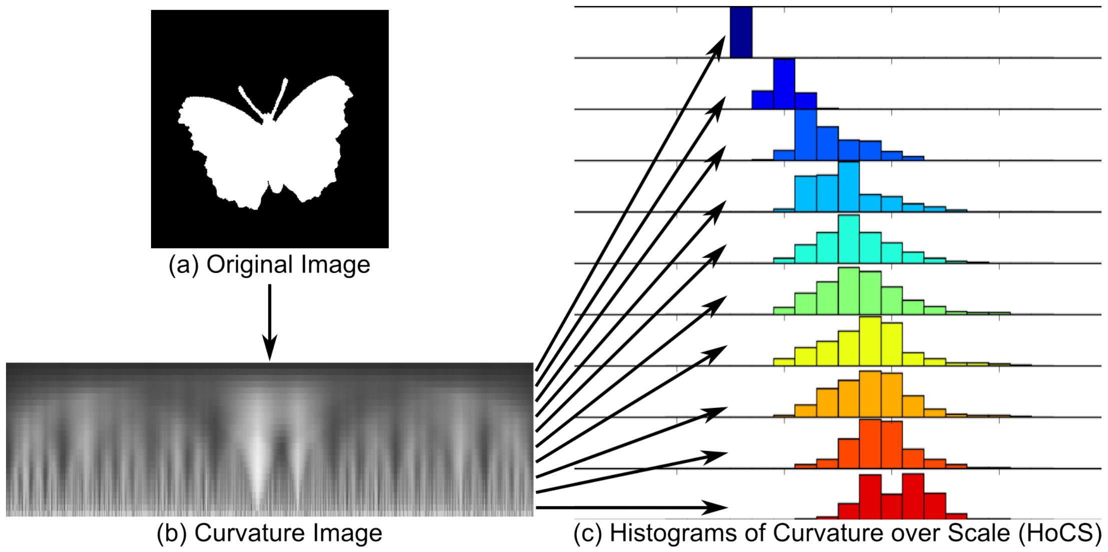


Figure 8.3: From (a) the original contour, we extract (b) the curvature image, which shows curvatures for each contour point along the x-axis and at different scales along the y-axis. (c) By taking histograms along each row, we create the *Histograms of Curvature over Scale* HoCS feature.

to check for each contour point – an order of magnitude improvement over a naive $4r^2$ check.

8.1.2 Histogram of Curvatures over Scale (HoCS) features

From a given contour, we can extract a *curvature image* as shown in Fig. 8.3, using one of the integral measures defined in the previous section. Each row in this image represents curvature values at a given scale; each column represents curvatures at all scales for a given contour point. Note that different images – even of the same shape – could have different contour lengths, and the alignment between them (the starting position from which to compare curvature values) is unknown. Thus, we cannot simply compare these curvature images directly. Even if we do an exhaustive search for a good alignment (as done by Manay, *et al*. [Manay *et al.*, 2006]), articulation and discretization artifacts can cause serious problems in matching. However, by taking histograms of curvatures, we largely sidestep these issues.

As shown in Fig. 8.3, we compute histograms of curvature values at each scale (*i.e.*, from each row of the curvature image) and then concatenate these histograms to form the

HoCS feature. This feature is insensitive to alignment, discretization artifacts and small articulations, and yet is quite discriminative, as demonstrated in Fig. 8.1.

8.2 Attributes

There are two stages in the identification process at which we would like to apply attributes:

1. Before segmentation, and
2. After low-level feature extraction.

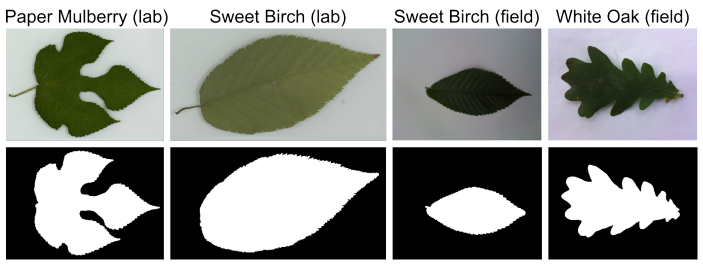
Item (1) is to reduce the number of computations and also to perhaps tune the segmentation/feature extraction steps on the basis of the computed attributes. Item (2) is to construct additional inputs to use during the classification step, or for changing the style of classification function used.

For the pre-segmentation attribute classifiers, we use the Gist feature [Oliva and Torralba, 2001] computed on the image as the low-level feature, and Support Vector Machines (SVMs) with an RBF kernel as the classification function. To ensure that the Gist values are scale-invariant, we resize the input input image to $320x240$ (rotating it by $90°$ if it has the wrong aspect ratio). We use the libsvm [Chang and Lin, 2001] implementation of SVMs. For the post-feature extraction attribute classifiers, we use all the HoCS features computed as inputs, and again use SVMs with RBF kernels as the classification function.

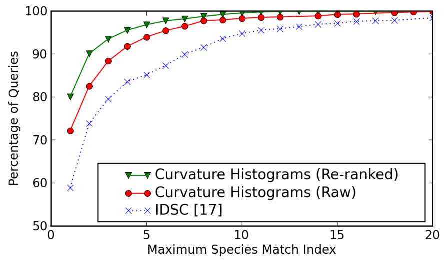
8.3 Experiments

Retrieval is done on segmented images, where contours are extracted and processed into curvature histograms. Feature dimensionalities are 31 scales $\times$ 33 bins per scale $=$ 1023 values for histograms of the area and arclength measures (each). Histograms are compared using the L_1 metric.

We evaluate on a dataset of real-world images used for plant species identification. This dataset consists of $4,162$ “clean” lab images of pressed leaves, taken with a high-quality camera, and 876 field images taken with different mobile devices. The field images, unlike the lab ones, contain varying amounts of blur and were photographed with different



(a) Leaf Images and Segmentations



(b) Performance Plot

Figure 8.4: (a) Leaf images and their segmentations. (b) Performance plot showing the percentage of queries with given match species index, which is defined as 1 plus the number of incorrect species with higher ranks than the first correctly matched species.

viewpoints and illumination, all of which affect the segmentation results. Examples of lab and field images, and their segmentations, are shown in Fig. 8.4a.

The experiment we perform is leave-one-image-out species identification, using the field images as queries. The goal is to have the lowest possible *species match index* for each query, which is matched against all other images (4,162 lab images + 875 remaining field images). The species match index is defined as 1 plus the number of incorrect species rated more similar to the query than the first correct match. So if a query image of class A has results of classes B, C, C, B, B, A (in that order), then the species index is $1 + 2 = 3$.

We use our HoCS feature, computed using both the area and arclength measures, with simple L_1 comparisons. A plot of recognition rates as a function of species match index across all queries is shown in Fig. 8.4b. Performance for IDSC [Ling and Jacobs, 2007] is shown as the dotted blue line, and for our curvature histogram features as the solid red line. 96.8% of queries have a species match index of 5 or lower with our method, which is substantially better than IDSC's 85.14%. This vast improvement highlights the effectiveness of our approach for shape retrieval on real-world images.

8.4 Leafsnap System

We have implemented our Leafsnap system as a back-end server that accepts recognition requests and various front-end apps for different devices that send requests. Currently, we have front-ends for the iPhone and iPad devices, with work on Android devices in progress.

Statistic	**Value**
Downloads	500,000
Users Attempted Recognition	173,265
Uploads	402,384
Valid Leaf Uploads	246,379
% Valid Leaves	61.2%
Labeled Leaves	18,390

Table 8.1: Various statistics about our Leafsnap system.

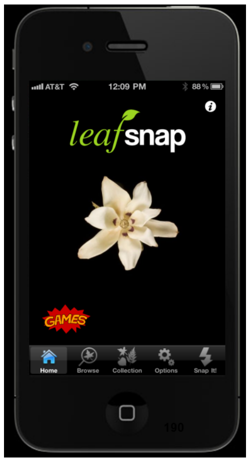
The iPhone app was launched on May 2, 2011, and the iPad app three weeks later. We received extensive press coverage from all major news outlets, including print newspapers (The New York Times and the Washington Post), wire services (the Associated Press and Reuters), magazines (Science), online blogs (Wired and Techcrunch), and many other press sources throughout the world. Together, the apps have been downloaded almost half a million times, and roughly one-third of the downloaders have actually tried submitting a leaf for recognition. Over $400,000$ images have been uploaded, of which about 60% are valid leaf images (*i.e.*, they passed our leaf/non-leaf classifier). These and other statistics are compiled in Table 8.4.

The server is currently a single Intel Xeon machine with 2 quad-core processors running at 2.33 Ghz each, and 4 GB of RAM. Aside from high-resolution versions of the Finding Species images, which are served via Amazon's "Simple Storage Service (S3)," all other operations are handled by our server.

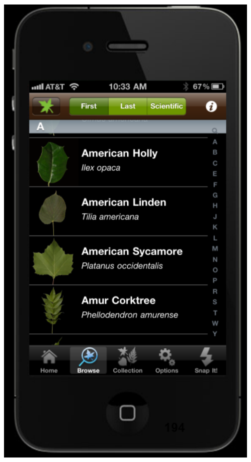
The iPhone and iPad apps are quite similar apart from the visual layout of screens, which are optimized for each device's screen real-estate. The major differences between the versions are:

- The iPhone version includes some games for learning to identify the different species.
- The iPad version includes a "nearby species" tab which allows users to see which species have been collected and labeled close to their current location.

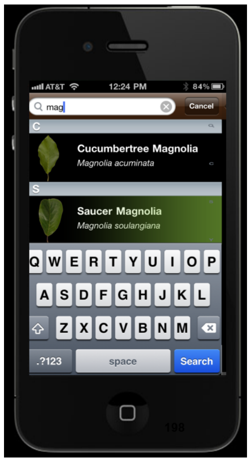
A "tour" of the iPhone app is shown in Fig. 8.5. In order from left-to-right, top-to-



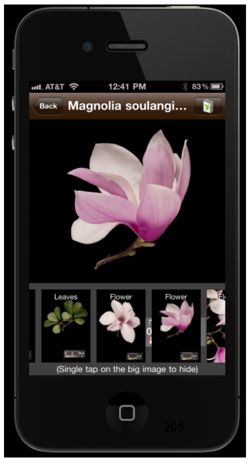
(a) Home



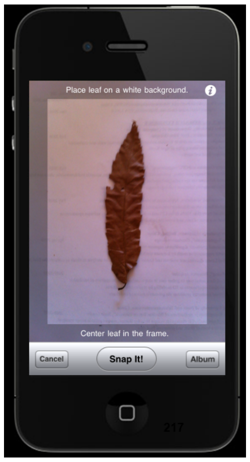
(b) Browse



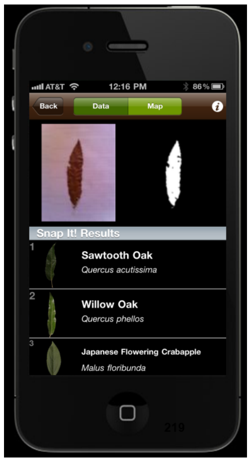
(c) Search



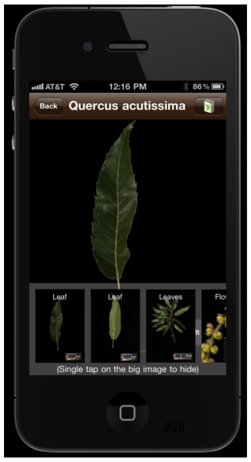
(d) Detail



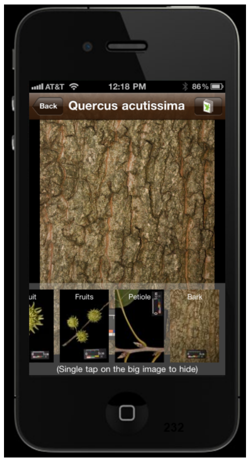
(e) Snap It!



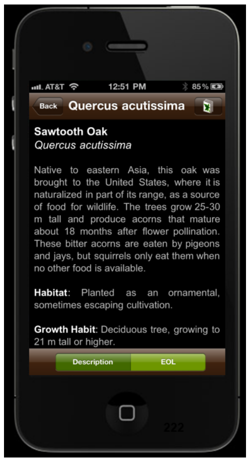
(f) Snap It! Results



(g) Verification 1



(h) Verification 2



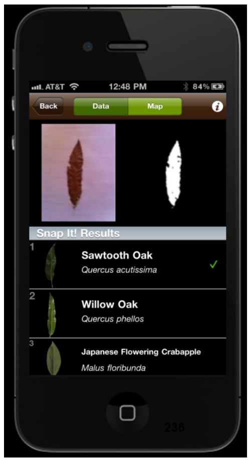
Sawtooth Oak
Quercus acutissima

Native to eastern Asia, this oak was brought to the United States, where it is naturalized in part of its range, as a source of food for wildlife. The trees grow 25-30 m tall and produce acorns that mature about 18 months after flower pollination. These bitter acorns are eaten by pigeons and jays, but squirrels only eat them when no other food is available.

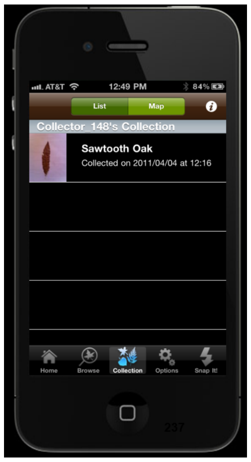
Habitat: Planted as an ornamental, sometimes escaping cultivation.

Growth Habit: Deciduous tree, growing to 21 m tall or higher.

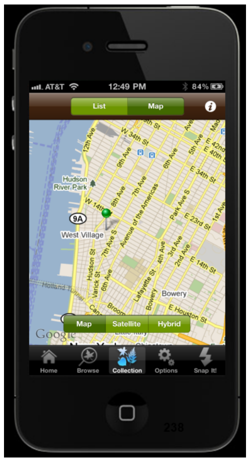
(i) Verification 3



(j) Label



(k) Collection



(l) Map

Figure 8.5: Tour of the iPhone version of Leafsnap. See text for details.

bottom: (a) the home screen, with a randomly-chosen image cycling every few seconds and access to the games, (b) the browse screen, with a sortable and searchable list of all the species contained in the guide, (c) the search functionality for finding particular species by scientific or common name, (d) the detail view for a particular species, showing the different images available for viewing, (e) the Snap It! screen, for when the user wants to perform automatic identification, (f) the returned identification results, in sorted order, (g-i) the manual verification stage as the user explores the images and textual descriptions of one of the results to confirm it as the right match, (j) labeling the right match, (k) showing the addition of that leaf to the user's collection for future reference, and (l) a map view showing where that leaf was collected.

8.5 Discussion

We have shown how to apply the concept of attribute classifiers to a domain other than faces – plant identification. At a low-level, we have shown how to efficiently compute curvature histograms at multiple scales on segmented leaf images. The power of these features comes from the careful computation of curvatures on discrete domains using integral measures, which produces reliable curvature values. By using histograms, we avoid the need for alignment and can handle small differences in shapes. These features are then used both directly for recognition, and also as inputs to a variety of attribute classifiers, which further boost recognition performance.

We believe that these curvature histogram features are particularly suited for shapes with variations at multiple scales, especially where segmentation or natural intra-class variations can cause local distortions in the shape. Our method can also handle images with multiple pieces, as histograms can simply be aggregated over all pieces. Analogoues of this feature can also be used in 3D, and could allow for many different research directions, including object recognition using shapes constructed from multiple segmentations of images, retrieval of 3D medical images (particularly those with volumetric representations), and 2D-based retrieval of 3D shapes (by looking at which 2D curvature histograms can be "contained" within a 3D curvature histogram).

On a more practical level, we have built and publicly released the Leafsnap system to the general public, demonstrating the effectiveness of our recognition system on real-world images of leaves. The system is alive and growing, as more people continue to download it and we continue to improve its operation and coverage. In the future, it is tempting to consider what other domains we could tackle similarly: Bugsnap? Cloudsnap? Dogsnap? The possibilities are almost limitless.

Chapter 9

Conclusions

In this thesis, we have shown how to automatically train classifiers for describable aspects of visual appearance – attributes and similes. These classifiers are learned using large collections of labeled images obtained from the internet. We demonstrated the use of these describable attributes for performing image search, automatic face replacement, and face verification. For search, we showed that by using our attribute-based FaceTracer engine, we were able to get significantly more relevant results to users' queries. Our automatic face replacement system was shown to be useful for a variety of tasks including privacy protection, de-identification, personalized replacement, and for creating pleasing composite group portraits. Finally, our face verification system using attributes showed performance comparable to the state-of-the-art, and the possibility of doing far better in the future as the underlying attribute classifiers are improved. We have also made available two large and complementary datasets for use by the community to make further progress along these lines.

We also demonstrated the use of visual attributes in a completely different domain – plant species identification. We developed this into our Leafsnap system, which we have released publicly to great acclaim.

These seem to be promising first steps in a new direction, and there are many avenues to explore. The experiments with human attribute labeling suggest that adding more attributes and improving the attribute training process could yield great benefits for face verification. Another direction to explore is how best to combine attribute and simile classifiers with low-level image cues. Finally, an open question is how attributes can be applied to domains other than faces. It seems that for reliable and accurate attribute training, analogues to the detection and alignment process must be found.

9.1 Metamers and Dynamic Selection of Attributes to Label

The set of attributes used in this work were chosen in an ad-hoc way; how to select them dynamically in a more principled manner is an interesting topic to consider. In particular, a system with a user-in-the-loop could be used to suggest new attributes. Thanks to Amazon Mechanical Turk, such a system would be easy to setup and could operate autonomously.

The objective function above can be used to identify deficiencies in the information provided by an initial set of attributes. The task is then to determine what additional attributes and/or training instances would improve the objective function, increasing mutual information and efficiency. It is important to note that we do not have to approach this problem blindly by considering each possible attribute, labeling instances for that attribute, and re-evaluating the objective function. Instead, we can take a short-cut by asking for human intervention. For instance we might have two instances, X and Y, with similar attribute values, $A(X) \approx A(Y)$, but different labels, $l(X) \neq l(Y)$. We call these metamers, following the nomenclature for colors that appear the same to a human observer despite differences in spectra. A person could be asked to suggest additional attributes that distinguish the two instances. A version of this has been recently described [Parikh and Grauman, 2011].

9.2 Attribute Dependencies

Another issue to tackle is identifying and effectively dealing with dependencies between attributes, a crucial issue that has not been solved in previous work (including our own) [Kumar *et al.*, 2008; Kumar *et al.*, 2009; Farhadi *et al.*, 2009; Lampert *et al.*, 2009]. We must deal with the fact that we do not have all attributes labeled for each training instance and that the correlations between attributes are not known *a priori*. One possibility is to label all attributes for some subset of data, identify dependent attributes using techniques like stabilized lasso [Bach, 2008], or more general feature selection approaches [Guyon and

Elisseeff, 2003], and then use structured prediction [Lafferty *et al.*, 2001; Taskar *et al.*, 2003]. Another direction of inquiry could be to formulate learning multiple attributes as a multi-task learning problem [Baxter, 1995; Caruana, 1997]. In this setting, some previous work may point toward techniques to handle training data items labeled only for a subset of tasks [Jebara, 2004]. An omnipresent concern when dealing with predicting so many correlated variables is bias in the training set. Recent results in information theory on optimal denoising [Weissman *et al.*, 2005] and their application to vision [Lazebnik and Raginsky, 2009] open up the possibility of using techniques that adapt to test data sampled from different distributions than the training data. We hope to modify this approach to attribute prediction.

Since the publication of work described in this thesis, attributes have exploded in popularity. Some of these include further progress on using attributes for object and category classification [Wang and Forsyth, 2009; Farhadi *et al.*, 2010a; Wang and Mori, 2010; Yu and Aloimonos, 2010; Torresani *et al.*, 2010; Russakovsky and Fei-Fei, 2010; Li-Jia Li and Fei-Fei, 2010b; Sadhegi and Farhadi, 2011; Hwang *et al.*, 2011; Siddiquie *et al.*, 2011; Douze *et al.*, 2011; Li-Jia Li and Fei-Fei, 2010a], better ways to collect data and attribute labels [Berg *et al.*, 2010b; Berg *et al.*, 2010a; Parikh and Grauman, 2011], using attribute classifiers to automatically generate descriptions of images [Farhadi *et al.*, 2010b; Kulkarni *et al.*, 2011], predicting aesthetics of images [Dhar *et al.*, 2011], activity recognition [Liu, 2011], and pose tracking [Sigal *et al.*, 2010]. Also related to attributes are the poselets of Bourdev *et al.* [Bourdev and Malik, 2009; Bourdev *et al.*, 2010; Brox *et al.*, 2011; Maji *et al.*, 2011], which have shown great promise for recognizing human poses, in addition to applications in segmentation, activity recognition, and other tasks as well.

Wherever the path of attributes leads us to, we have several exciting years ahead!

Appendix A

Mechanical Turk Usage

We label our training data using Amazon's Mechanical Turk (MTurk) service [Amazon, 2011], which allow us to create online "Human Intelligence Tasks" (HITs). Workers who have registered with Mechanical Turk can sign on and complete the HITs. Our HITs consist of web pages that show some images and ask the worker for some sort of judgement. For example we show two faces and ask if they show the same person, to build our verification training data. Or we show thirty faces and ask the worker to select all of them with blond hair, for our attribute classifier training data.

Here are the different types of labeling jobs we've done using Mechanical Turk:

1. Attribute Labeling (Sec. A.3)
2. Identity Labeling (Sec. A.4)
3. Face Verification (Sec. A.5)
4. Attribute Verification for boosting (Sec. A.6)
5. Joint Attribute Labeling for bias removal (Sec. A.7)

Below is some detailed information about how to get started with Mechanical Turk yourself.

A.1 Mechanical Turk Jobs

A job in Mechanical Turk is defined using 3 pieces:

1. **A page template:** This consists of HTML code that defines what a worker sees, with placeholders for template arguments, such as images, *etc.* This takes any standard HTML code, including CSS stylesheets for custom styling and javascript scripts for adding interactive and intuitive user-interface functionality, or even links to external databases, etc. This is a very flexible system, and as more applications move to the web, it means that this system will be increasingly familiar to software developers of all types.

2. **Batch data file:** A batch of data to fill into the template, given as a Comma-Separated Values (CSV) file. All the variables defined in the page template are filled in here, for as many individual HITs as you want to create. It's easy to create these from any programming language, or even Excel. You can also have as many lines (*i.e.*, HITs) as you want within a single CSV file, allowing for the creation of small and large batches alike.

3. **Additional metadata:** Some metadata about the job, such as keywords, price, quality filters, *etc.* These are used to determine how workers can find your job, which workers are eligible to work on it (*i.e.*, they must have a certain approval rating on past jobs), how much they get paid, how much time they have available to work on each job, how many individual workers must complete each job, *etc.* You set this once per job, and it's easy to modify if you want to experiment with different prices, *etc.*

You can create as many templates as you want, and you can upload as many CSVs as you want. The cost of running a job is set by you, and Amazon takes an additional 10% or \$0.005 per HIT, whichever is higher.

Once you submit a job, there's a progress page showing how fast workers are completing the HITs. You can even view results as a table while in progress. Once done, you can export the results as a CSV file to analyze them as you wish.

Workers are paid either automatically after a time limit (default 1 week), or you can explicitly approve/reject individual HITs. This means that if you find workers who are sloppy or cheating, you can reject them (they are not paid) and additionally ban them forever from working on your jobs.

A.2 Strategies and Tips

Over the past few years that we have been using MTurk, we have discovered several important things about how to most effectively use it:

- Jobs should typically take between 30 seconds to 2 minutes. There are many more workers available to take these kind of jobs than longer ones. The typical rate is anywhere from 1 to 5 cents for such jobs. However, with such low rates, Amazon's fees are proportionately higher.
- The utmost care should be taken to streamline the page templates for the workers. Minimize scrolling, make instructions very clear and as explicit as possible, make anything that they have to click as big as possible, use colors to make things clearer and provide feedback on state changes, use some javascript if needed to make things easier, provide adequate feedback for what a completion means, *etc.*
- If you need things done faster, raising the price helps a lot!
- Keep in mind that some workers do hundreds or thousands of our HITs – and they don't re-read the instructions every time. If you change a task in a subtle way, workers are likely not to notice and continue doing the old task.
- A tedious and painstaking job (*e.g.*, Fiducial Labeling) makes doing bad work very attractive for the worker – at least relative to doing good work. Some quality control becomes necessary. Something to remember in general is that there are two types of bad work: honest bad work, where the worker made an attempt to complete the task but either didn't understand it or wasn't sufficiently careful; and spam, where the worker did the minimum required to enable the submit button, without regard

to the task. Spam, if found, can be rejected (not paid for), and the worker blocked from our future HITs. Honest bad work is worse: It's harder to find, it's more likely to corrupt our data (because it's harder to find and ignore), and rejecting it is more likely to give us a bad reputation. Here are some points about HITs that garner a lot of bad work.

– Manual review is worthwhile for some jobs. It's a pain, but made easier by sorting the results before looking through them. Group by worker, and sort by number of jobs completed – the workers who do a lot of jobs tend to be the worst both in quality (because they're just speeding through the HITs) and in damage done (because each bad worker corrupts a larger portion of the results). The largest fiducial labeling job we ran consisted of 1312 tasks, each to be done by 8 workers. The most prolific worker did 1013 of the tasks all of which (that we checked) were spam. The 8 most prolific workers did half of the assignments. Just looking through a few workers can rid you of most of your bad data.

– We've been reluctant to block workers who do honest bad work because of the consequences (workers blocked by several requesters will be kicked out of MTurk entirely). But these workers can actually do more damage than spammers. A way around this is by using qualifications:

   1. Create a qualification which is automatically granted upon request (this type of qualification cannot be created through the web UI, but can be done with a script through the REST API. Give it a default value of 50. Require this qualification with a value of at least 40 to do your HITs. We don't have a lot of data to support this, but it seems like even an automatically granted qualification makes workers more careful – or scares away the careless.
   2. Decrease the score of workers who do bad work, and increase it for workers who do good work. If you have a way of stepping through results and flagging them as good or bad, it's pretty easy to write a script to update workers' scores, and even send them messages with feedback.
   3. Although we haven't taken it this far yet, if you build up a pool of known

good workers this way, you may be able to run future HITs with fewer repetitions per task, by requiring a higher qualification score (possibly paying more for these, making workers value the qualification).

A.3 Attribute Labeling

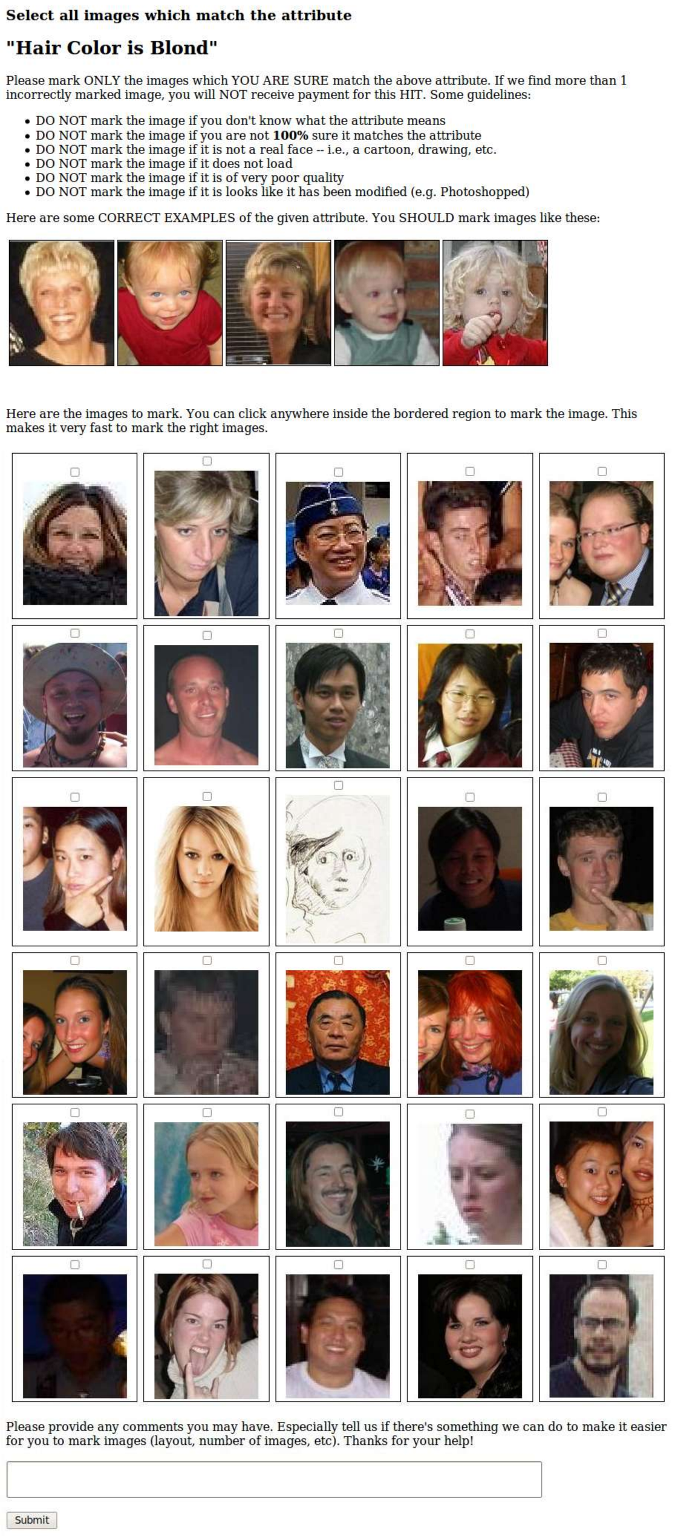
We collect training data for the attribute classifiers by posting MTurk jobs like that shown in Fig. A.1. On each job, workers are shown 30 cropped face images from our list of faces and asked to click the faces which match the given attribute. They're asked to be conservative; *i.e.*, to only mark images for which they're sure. This way, we get a minimum of junk labels. For binary attributes, we submit separate jobs for the positive and negative cases.

One trick we can use when creating jobs is to look at the faces not marked in previous jobs and select more faces from this set when creating a job for a competing attribute. For example, if we showed a page of faces asking for workers to click on male, and then we create a new job for labeling female faces, the ones that were not clicked in the first job are more likely to be female and hence we get more usable results in the second job by this sampling strategy.

The raw labels obtained by each batch are stored in a database table locally, and we then aggregate counts for each image (*i.e.*, across all the workers that have seen that image for that attribute) to get an overall confidence. If this confidence is high enough, then the image is considered a positive (or negative) example of the attribute, and used during feature selection/classifier training. The threshold for confidence is dependent on the number of times an image has been seen. It has to be seen at least 3 times, and agreed upon all 3 times to become a confirmed label. If an image is seen more than 3 times, then a sliding scale used.

A.4 Identity Labeling

We collect training data for building datasets of faces labeled by identity by posting jobs like that shown in Fig. A.2.



Select all images which match the attribute

"Hair Color is Blond"

Please mark ONLY the images which YOU ARE SURE match the above attribute. If we find more than 1 incorrectly marked image, you will NOT receive payment for this HIT. Some guidelines:

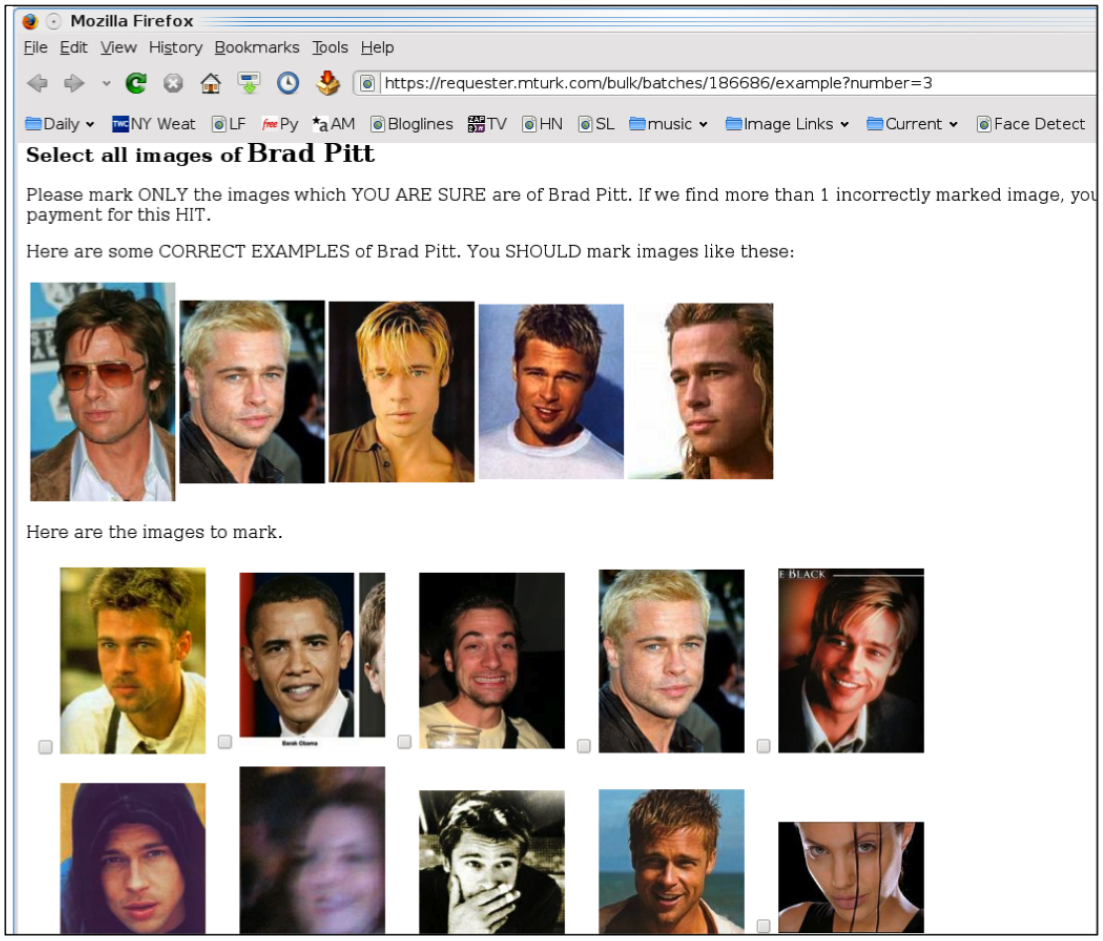
- DO NOT mark the image if you don't know what the attribute means
- DO NOT mark the image if you are not **100%** sure it matches the attribute
- DO NOT mark the image if it is not a real face -- i.e., a cartoon, drawing, etc.
- DO NOT mark the image if it does not load
- DO NOT mark the image if it is of very poor quality
- DO NOT mark the image if it is looks like it has been modified (e.g. Photoshopped)

Here are some CORRECT EXAMPLES of the given attribute. You SHOULD mark images like these:

Here are the images to mark. You can click anywhere inside the bordered region to mark the image. This makes it very fast to mark the right images.

Please provide any comments you may have. Especially tell us if there's something we can do to make it easier for you to mark images (layout, number of images, etc). Thanks for your help!

Figure A.1: An example of an attribute labeling job.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://requester.mturk.com/bulk/batches/186686/example?number=3
Daily NY Weat LF Py AM Bloglines TV HN SL music Image Links Current Face Detect

Select all images of Brad Pitt

Please mark ONLY the images which YOU ARE SURE are of Brad Pitt. If we find more than 1 incorrectly marked image, you
payment for this HIT.

Here are some CORRECT EXAMPLES of Brad Pitt. You SHOULD mark images like these:

Here are the images to mark.

Figure A.2: An example of an identity labeling job.

On each job, workers are shown 30 cropped face images from our list of faces (usually a subset that we expect to contain images of this celebrity) and asked to click the faces which are of the given person (named at the top, and with 5 example images shown). They're asked to be conservative; *i.e.*, to only mark images for which they're sure it's the same person. This way, we get a minimum of junk labels.

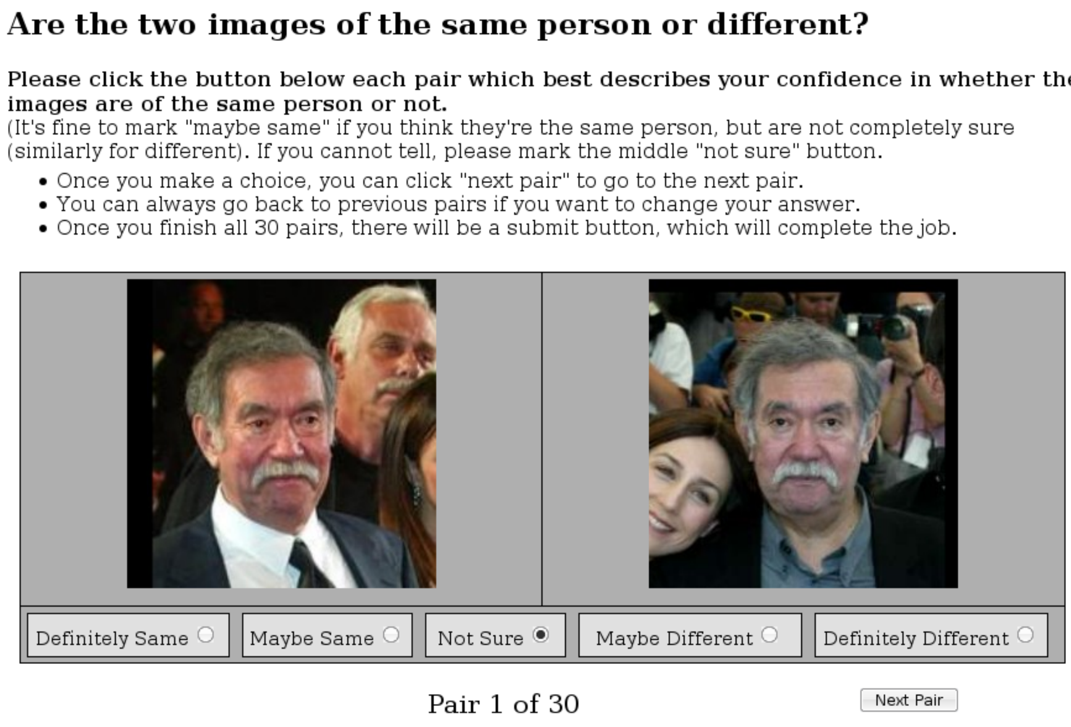
The raw labels obtained by each batch are stored in a database table locally, and we then aggregate counts for each image (*i.e.*, across all the workers that have seen that image for that attribute) to get an overall confidence. If this confidence is high enough, then the image is considered a positive image of this person.

We afterwards run duplicate-detection software to get rid of duplicates, conservatively.

We use the linux program `findimagedupes`[1] for this task.

A.5 Face Verification

Fig. A.3 shows an example of the human face verification task, as presented to workers on MTurk.



Are the two images of the same person or different?

Please click the button below each pair which best describes your confidence in whether the images are of the same person or not.
(It's fine to mark "maybe same" if you think they're the same person, but are not completely sure (similarly for different). If you cannot tell, please mark the middle "not sure" button.

- Once you make a choice, you can click "next pair" to go to the next pair.
- You can always go back to previous pairs if you want to change your answer.
- Once you finish all 30 pairs, there will be a submit button, which will complete the job.

Definitely Same ○ Maybe Same ○ [✓ Not Sure ●] Maybe Different ○ Definitely Different ○

Pair 1 of 30

Figure A.3: An example of a face verification job, used to evaluate human performance on the LFW benchmark [Huang *et al.*, 2007c].

Directions are given at the top, and then the two images are shown along with 5 different confidence options below the pair of images. A user *must* click on one of the options before they can go on to the next pair. The background color changes according to the user's choice, as a visual cue to let them know which option they picked. Each job consists of 30 pairs.

By averaging the responses from 10 different workers, we are able to get a real-valued

[1]`http://www.jhnc.org/findimagedupes/`

"score" for each verification pair, which we can then slide a threshold over to generate an ROC curve.

A.6 Attribute Verification for Boosting

A standard practice when training classifiers, boosting attempts to produce a new classifier that is better at predicting samples for which the current classifiers performance is poor. To do this, we first classify some of our images with our attribute classifier for the given attribute. We then submit images for a given attributes to Mechanical Turk, asking workers to click on images which don't match the given attribute. These jobs look like that shown in Fig. A.4. From the results of these jobs, we get a list of faces which our current classifiers are wrong on. We then include these faces more prominently in our feature selection and final training procedures.

Some statistics about the improvements made using boosting are given in Table A.1

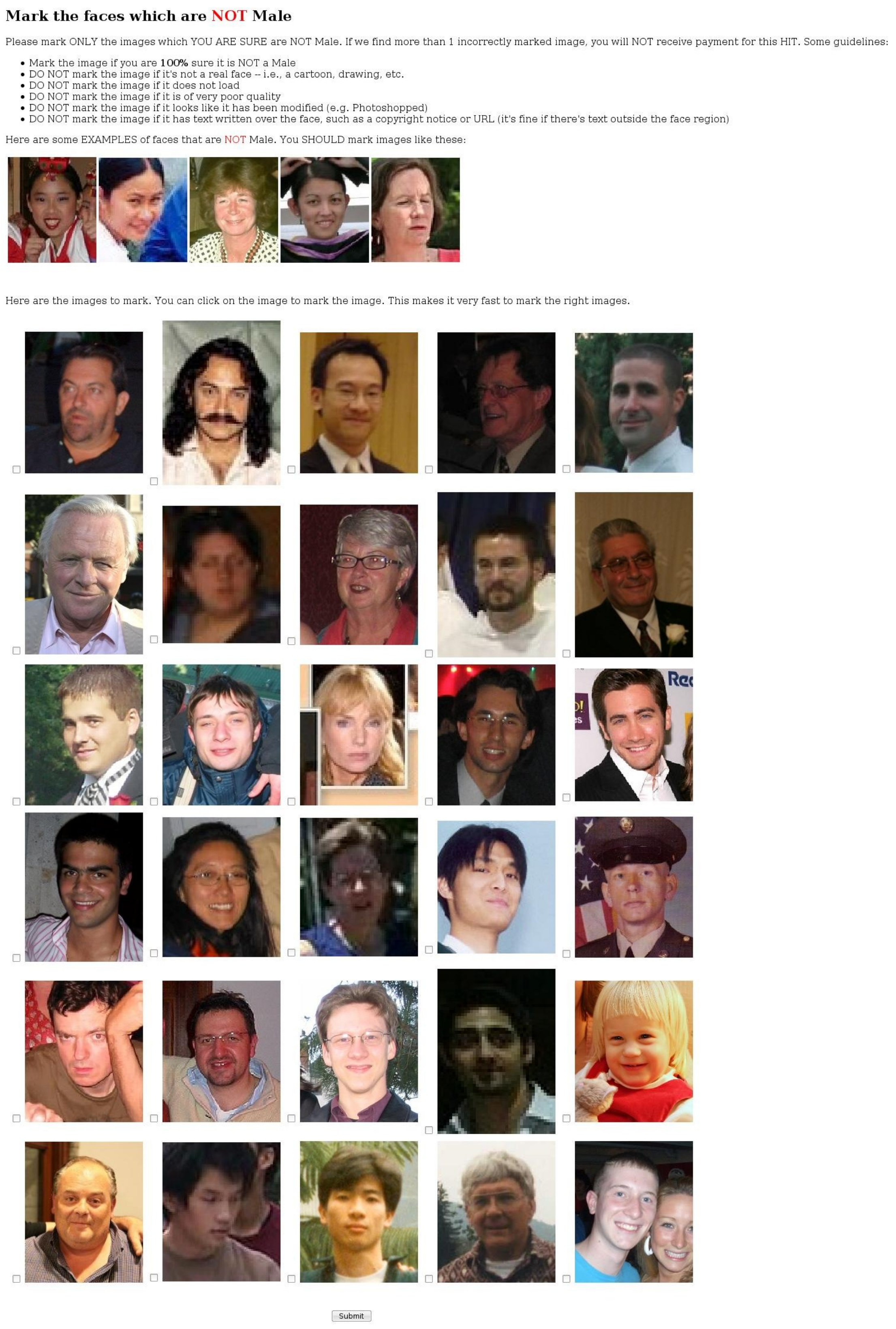
Attribute	**# Incorrect Classifications**	**Old Acc.**	**New Acc.**
Gender	1618	81.22%	85.78%
Asian	506	92.32%	93.80%

Table A.1: By emphasizing images which our current attribute classifiers err on, we are able to boost their performance, as shown here for the "gender" and "asian" attributes.

A.7 Joint Attribute Labeling for Bias Removal

Some attribute scores are highly correlated, for example "gender" and "eyebrow thickness" (see Fig. A.5a). One suspects that a correlation in attribute scores would be reflected as a correlation in MTurk labels, if we asked for labels for both these attributes on the same images. Fig. A.5b and c show 20 randomly chosen faces labeled as having bushy and thin eyebrows, respectively. If we had gender labels for these, it's pretty clear there would be a high correlation.

If the correlation is strong, and one of the attributes is difficult to learn, the classifier for that attribute may just mimic the classifier for the other attribute. For example, in



Mark the faces which are NOT Male

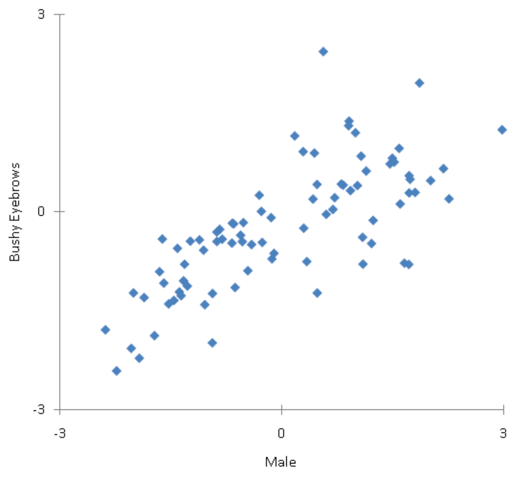
Please mark ONLY the images which YOU ARE SURE are NOT Male. If we find more than 1 incorrectly marked image, you will NOT receive payment for this HIT. Some guidelines:

- Mark the image if you are **100%** sure it is NOT a Male
- DO NOT mark the image if it's not a real face -- i.e., a cartoon, drawing, etc.
- DO NOT mark the image if it does not load
- DO NOT mark the image if it is of very poor quality
- DO NOT mark the image if it looks like it has been modified (e.g. Photoshopped)
- DO NOT mark the image if it has text written over the face, such as a copyright notice or URL (it's fine if there's text outside the face region)

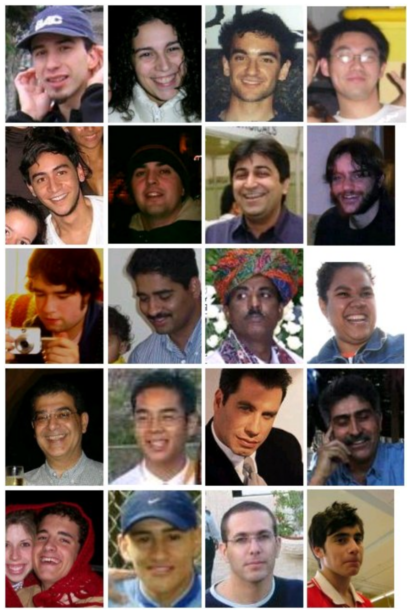
Here are some EXAMPLES of faces that are NOT Male. You SHOULD mark images like these:

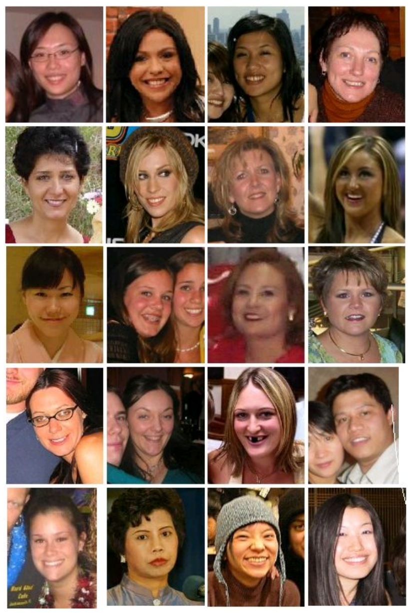
Here are the images to mark. You can click on the image to mark the image. This makes it very fast to mark the right images.

Figure A.4: An example of an attribute verification job.



(a) Gender and eyebrow thickness correlation





(b) Random "bushy eyebrow" images (c) Random "thin eyebrow" images

Figure A.5: (a) Some attributes are highly correlated, such as "gender" and "eyebrow thickness." This makes it more difficult to obtain good training data for the correlated attribute. Randomly chosen images labeled as (b) "bushy eyebrows" and (c) "thin eyebrows" show a strong correlation with gender. This causes the learned classifier to essentially mimic the gender classifier, regardless of actual eyebrow thickness.

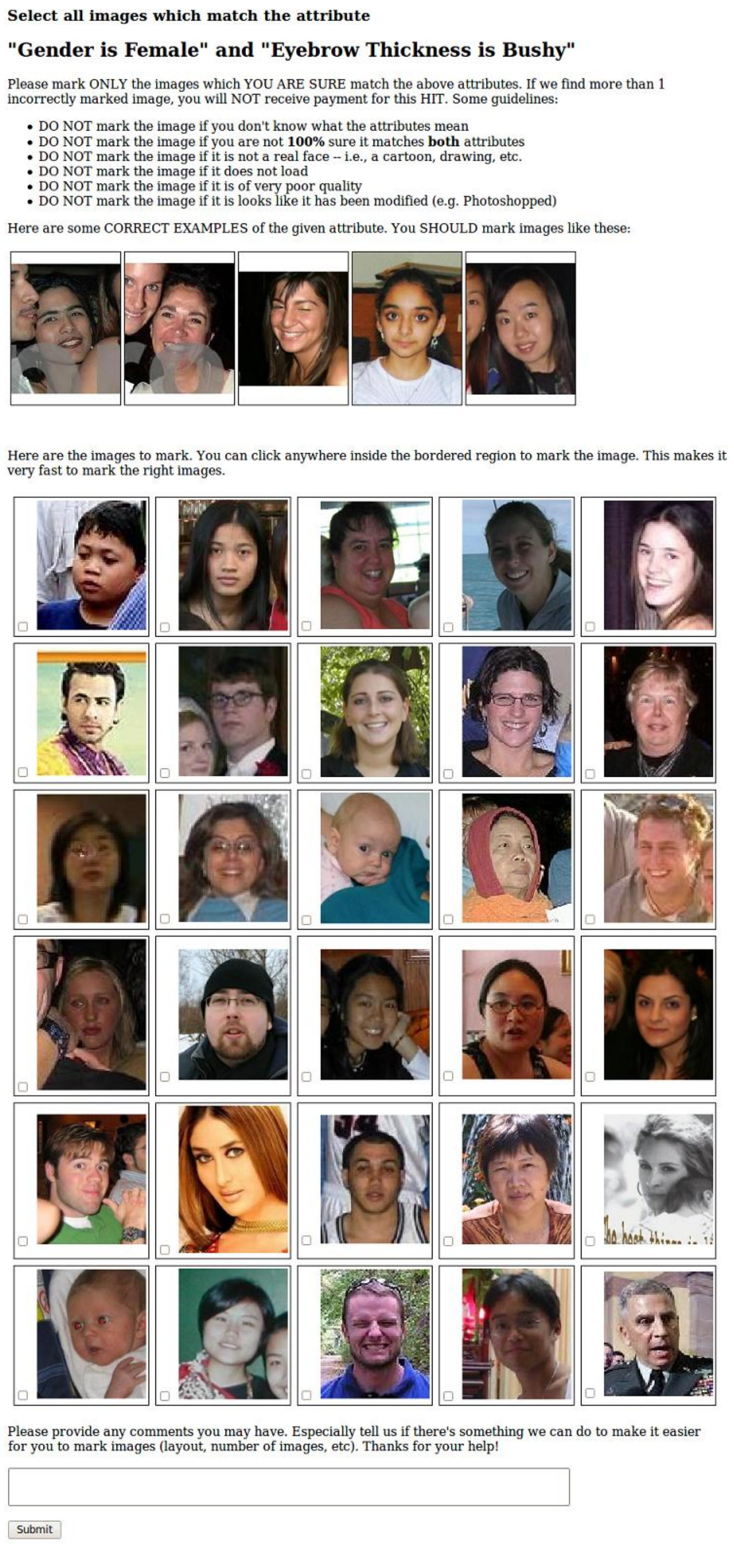
the gender-eyebrow case, if gender is easier to learn than eyebrow thickness, the nominal eyebrow thickness classifier may really be a gender classifier. If our end goal is eyebrow classification, this may not be a bad thing. But if our goal is to build a set of composable attribute classifiers, to be combined in a higher order classifier, it's important that each classifier bring as much new information to the party as possible.

To confirm this, we collected these labels for the "gender (male/female)" and "eyebrow thickness (bushy/thin)" attributes in a set of joint attribute labeling HITs, an example of which is shown in Fig. A.6. Some statistics on the data gathered are listed in Table A.2. We have at least 500 samples labeled with each of the four possible combinations of these two attributes. Holding out 100 samples from each of these four sets, we train on the remainder. This gives us a gender-balanced eyebrow thickness classifier, which does better than our previous eyebrow thickness classifier on the rare combinations, as can be seen in Table A.2.

Joint Label	**# Labeled**	**Hit Rate**	**Old Acc.**	**Balanced Acc.**
male, bushy eyebrows	650	10.6%	94%	91%
female, bushy eyebrows	503	2.9%	35%	70%
male, thin eyebrows	522	3.5%	62%	80%
female, thin eyebrows	638	9.3%	96%	95%
entire test set	2,313	7.9%	92%	92%

Table A.2: By submitting joint attribute labeling jobs, we are able to gather a balanced dataset of images for the "eyebrow thickness classifier." By retraining the eyebrow thickness classifier using this balanced set, we obtain a significant boost in performance for the rarer cases of males with thin eyebrows and females with thick eyebrows.

Notice how awful the old classifier is on the rare subset. Improving accuracy on the rare subsets does not improve accuracy overall (the common case becomes slightly worse), but the new classifier may be less correlated with the gender classifier, and so provide more information to the higher level verification classifier.



Select all images which match the attribute

"Gender is Female" and "Eyebrow Thickness is Bushy"

Please mark ONLY the images which YOU ARE SURE match the above attributes. If we find more than 1 incorrectly marked image, you will NOT receive payment for this HIT. Some guidelines:

- DO NOT mark the image if you don't know what the attributes mean
- DO NOT mark the image if you are not **100%** sure it matches **both** attributes
- DO NOT mark the image if it is not a real face -- i.e., a cartoon, drawing, etc.
- DO NOT mark the image if it does not load
- DO NOT mark the image if it is of very poor quality
- DO NOT mark the image if it is looks like it has been modified (e.g. Photoshopped)

Here are some CORRECT EXAMPLES of the given attribute. You SHOULD mark images like these:

Here are the images to mark. You can click anywhere inside the bordered region to mark the image. This makes it very fast to mark the right images.

Please provide any comments you may have. Especially tell us if there's something we can do to make it easier for you to mark images (layout, number of images, etc). Thanks for your help!

Figure A.6: An example of an joint attribute labeling job, which helps reduce correlation bias.

Bibliography

[Agarwal *et al.*, 2006] G. Agarwal, P. Belhumeur, S. Feiner, D. Jacobs, W.J. Kress, R. Ramamoorthi, N. Bourg, N. Dixit, H. Ling, D. Mahajan, R. Russell, S. Shirdhonkar, K. Sunkavalli, and S. White. First steps toward an electronic field guide for plants. *Taxon, Journal of the International Association for Plant Taxonomy*, 55(3):597–610, 2006.

[Agarwala *et al.*, 2004] Aseem Agarwala, Mira Dontcheva, Maneesh Agrawala, Steven Drucker, Alex Colburn, Brian Curless, David Salesin, and Michael Cohen. Interactive Digital Photomontage. *ACM Transactions on Graphics*, 23:294–302, 2004.

[Amazon, 2011] Amazon. Mechanical turk. `http://mturk.com`, 2011.

[Avidan and Shamir, 2007] Shai Avidan and Ariel Shamir. Seam carving for content-aware image resizing. *ACM Transactions on Graphics*, 26, 2007.

[Bach, 2008] F. R. Bach. Bolasso: model consistent lasso estimation through the bootstrap. In *ICML*, 2008.

[Baluja and Rowley, 2007] Shumeet Baluja and Henry Rowley. Boosting sex identification performance. *Intl. Journal of Computer Vision (IJCV)*, 2007.

[Basri and Jacobs, 2003] R. Basri and D.W. Jacobs. Lambertian reflectance and linear subspaces. *IEEE TPAMI*, 25:218–233, 2003.

[Baxter, 1995] J. Baxter. Learning internal representations. In *CoLT*, 1995.

[Belhumeur *et al.*, 1996] P.N. Belhumeur, J. Hespanha, and D.J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. In *European Conf. on Computer Vision (ECCV)*, pages 45–58, 1996.

[Belhumeur *et al.*, 2008] P.N. Belhumeur, D. Chen, S. Feiner, D.W. Jacobs, W. John Kress, H. Ling, I. Lopez, R. Ramamoorthi, S. Sheorey, S. White, and L. Zhang. Searching the world's herbaria: A system for visual identification of plant species. In *ECCV*, 2008.

[Belongie *et al.*, 2002] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *TPAMI*, 2002.

[Berg *et al.*, 2004] T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Yee-Whye Teh, E. Learned-Miller, and D.A. Forsyth. Names and faces in the news. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[Berg *et al.*, 2010a] Tamara L. Berg, Alexander C. Berg, and Jonathan Shih. Automatic attribute discovery and characterization from noisy web data. In *European Conf. on Computer Vision (ECCV)*, 2010.

[Berg *et al.*, 2010b] Tamara L. Berg, Alexander Sorokin, Gang Wang, David A. Forsyth, Derek Hoiem, Ali Farhadi, and Ian Endres. It's all about the data. volume 98, pages 1434–1453, 2010.

[Bitouk *et al.*, 2008] Dmitri Bitouk, Neeraj Kumar, Samreen Dhillon, Peter N. Belhumeur, and Shree K. Nayar. Face swapping: Automatically replacing faces in photographs. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, August 2008.

[Blanz *et al.*, 2002] V. Blanz, S. Romdhani, and T. Vetter. Face identification across different poses and illuminations with a 3d morphable model. In *IEEE Conf. on Automatic Face and Gesture Recognition*, 2002.

[Blanz *et al.*, 2004] Volker Blanz, Kristina Scherbaum, Thomas Vetter, and Hans-Peter Seidel. Exchanging Faces in Images. *Computer Graphics Forum*, 23:669–676, 2004.

[Bourdev and Malik, 2009] Lubomir Bourdev and Jitendra Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *IEEE Intl. Conf. on Computer Vision (ICCV)*, 2009.

[Bourdev *et al.*, 2010] Lubomir Bourdev, Subhransu Maji, Thomas Brox, and Jitendra Malik. Detecting people using mutually consistent poselet activations. In *European Conf. on Computer Vision (ECCV)*, 2010.

[Boyle *et al.*, 2000] Michael Boyle, Christopher Edwards, and Saul Greenberg. The Effects of Filtered Video on Awareness and Privacy. In *ACM Conference on Computer Supported Cooperative Work*, 2000.

[Branson *et al.*, 2010] Steve Branson, Catherine Wah, Boris Babenko, Florian Schroff, Peter Welinder, Pietro Perona, and Serge Belongie. Visual recognition with humans in the loop. In *European Conference on Computer Vision (ECCV)*, Heraklion, Crete, Sept. 2010.

[Brox *et al.*, 2011] Thomas Brox, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Object segmentation by alignment of poselet activations to image contours. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Bruce *et al.*, 1999] V. Bruce, Z. Henderson, K. Greenwood, P. J. B. Hancock, A. M. Burton, and P. I. Miller. Verification of face identities from images captured on video. *Journal of Experimental Psychology: Applied*, 5:339–360, 1999.

[Burton *et al.*, 1999] A. Mike Burton, Stephen Wilson, Michelle Cowan, and Vicki Bruce. Face recognition in poor-quality video: Evidence from security surveillance. *Psychological Science*, 10(3):243–248, 1999.

[Caruana, 1997] R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

[Castillo and Jacobs, 2007] Carlos D. Castillo and David W. Jacobs. Using stereo matching for 2-d face recognition across pose. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[Chang and Lin, 2001] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[Chang *et al.*, 2005] Shih-Fu Chang, Winston Hsu, Lyndon Kennedy, Lexing Xie, Akira Yanagawa, Eric Zavesky, and Dongqing Zhang. Columbia university trecvid-2005 video search and high-level feature extraction. In *NIST TRECVID workshop*, Gaithersburg, MD, November 2005.

[Chang *et al.*, 2006] Shih-Fu Chang, Winston Hsu, Wei Jiang, Lyndon Kennedy, Dong Xu, Akira Yanagawa, and Eric Zavesky. Columbia University TRECVID-2006 Video Search and High-Level Feature Extraction. In *NIST TRECVID Workshop*, Gaithersburg, MD, November 2006.

[Chang *et al.*, 2007] Shih-Fu Chang, Dan Ellis, Wei Jiang, Keansub Lee, Akira Yanagawa, Alexander C. Loui, and Jiebo Luo. Large-scale multimodal semantic concept detection for consumer video. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, MIR '07, pages 255–264, New York, NY, USA, 2007. ACM.

[Chen *et al.*, 2000a] H. F. Chen, P. N. Belhumeur, and D. W. Jacobs. In search of illumination invariants. *CVPR*, 2000.

[Chen *et al.*, 2000b] H.F. Chen, P.N. Belhumeur, and D.W. Jacobs. In search of illumination invariants. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2000.

[Connolly, 1986] M L Connolly. Measurement of protein surface shape by solid angles. *Journal of Molecular Graphics*, 4(1):3 – 6, 1986.

[Cootes *et al.*, 2000] T.F. Cootes, K. Walker, and C.J. Taylor. View-based active appearance models. In *IEEE Conf. on Automatic Face and Gesture Recognition*, 2000.

[Cootes *et al.*, 2001] T. Cootes, G. Edwards, and C. Taylor. Active Appearance Models. *IEEE TPAMI*, 26:681–685, 2001.

[Cortes and Vapnik, 1995] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3), 1995.

[Cottrell and Metcalfe, 1990] Garrison W. Cottrell and Janet Metcalfe. Empath: face, emotion, and gender recognition using holons. In *Advances in Neural Information Processing Systems (NIPS)*, pages 564–571, 1990.

[Cox *et al.*, 2000] I.J. Cox, M.L. Miller, T.P. Minka, T.V. Papathomas, and P.N. Yianilos. The bayesian image retrieval system, pichunter: Theory, implementation and psychophysical experiments. *IEEE transactions on image processing*, 9:20–37, 2000.

[Dalal and Triggs, 2005] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005.

[Datta *et al.*, 2005] Ritendra Datta, Jia Li, and James Z. Wang. Content-based image retrieval: Approaches and trends of the new age. *Multimedia Information Retrieval*, pages 253–262, 2005.

[Debevec *et al.*, 2000] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the Reflectance Field of a Human Face. In *SIGGRAPH 00*, pages 145–156, 2000.

[Debevec, 1998] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *SIGGRAPH 98*, pages 189–198, 1998.

[Deng *et al.*, 2009] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[Dhar *et al.*, 2011] Sagnik Dhar, Vicente Ordonez, and Tamara L. Berg. High level describable attributes for predicting aesthetics and interestingness. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Douze *et al.*, 2011] Matthijs Douze, Arnau Ramisa, and Cordelia Schmid. Combining attributes and fisher vectors for efficient image retrieval. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Efros and Freeman, 2001] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH 01*, pages 341–346, August 2001.

[Everingham *et al.*,] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html.

[Everingham *et al.*, 2006] M. Everingham, J. Sivic, and A. Zisserman. Hello! my name is... Buffy – automatic naming of characters in TV video. In *British Machine Vision Conference (BMVC)*, 2006.

[Farhadi *et al.*, 2009] A. Farhadi, I. Endres, D. Hoiem, and D.A. Forsyth. Describing objects by their attributes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[Farhadi *et al.*, 2010a] A. Farhadi, I. Endres, and D. Hoiem. Attribute-centric recognition for cross-category generalization. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[Farhadi *et al.*, 2010b] A. Farhadi, Mohsen Hejrati, Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences for images. In *European Conf. on Computer Vision (ECCV)*, 2010.

[Ferencz *et al.*, 2007] A Ferencz, E Learned-Miller, and J Malik. Learning to locate informative features for visual identification. *Intl. Journal of Computer Vision (IJCV) Special Issue on Learning and Vision*, 2007.

[Fergus *et al.*, 2006] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam Roweis, and William Freeman. Removing camera shake from a single photograph. *SIGGRAPH 06*, pages 787–794, 2006.

[Ferrari and Zisserman, 2007] V. Ferrari and A. Zisserman. Learning visual attributes. In *Advances in Neural Information Processing Systems (NIPS)*, December 2007.

[Fisher, 1936] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[Flickner *et al.*, 1995] M. Flickner, H.S. Sawhney, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The qbic system. *IEEE Computer*, 28(9):23–32, 1995.

[Freund and Shapire, 1996] Y. Freund and R.E. Shapire. Experiments with a new boosting algorithm. In *Intl. Conf. on Machine Learning (ICML)*, 1996.

[Frome *et al.*, 2007] A. Frome, F. Sha, Y. Singer, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. *ICCV*, 2007.

[Gallagher and Chen, 2008] A.C. Gallagher and Tsuhan Chen. Estimating age, gender, and identity using first name priors. *CVPR*, 2008.

[Georghiades *et al.*, 2001] Athinodoros S. Georghiades, Peter N. Belhumeur, and David J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, 23(6):643–660, 2001.

[Golomb *et al.*, 1990] B. A. Golomb, D. T. Lawrence, and T. J. Sejnowski. SexNet: A neural network identifies sex from human faces. In *Advances in Neural Information Processing Systems (NIPS)*, pages 572–577, 1990.

[Griffin *et al.*, 2007] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[Gross *et al.*, 2001] R. Gross, J. Shi, and J. Cohn. Quo vadis face recognition? In *Workshop on Empirical Evaluation Methods in Computer Vision*, December 2001.

[Gross *et al.*, 2006] Ralph Gross, Latanya Sweeney, Fernando de la Torre, and Simon Baker. Model-Based Face De-Identification. pages 161–168, 2006.

[Guyon and Elisseeff, 2003] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *JMLR*, 3, 2003.

[Hadjidemetriou *et al.*, 2004] E. Hadjidemetriou, M.D. Grossberg, and S.K. Nayar. Multiresolution Histograms and their use for Recognition. *TPAMI*, 26(7):831–847, Jul 2004.

[Hays and Efros, 2007] James Hays and Alexei A Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics*, 26(3), 2007.

[Hua and Akbarzadeh, 2009] Gang Hua and A. Akbarzadeh. A robust elastic and partial matching metric for face recognition. In *IEEE Intl. Conf. on Computer Vision (ICCV)*, pages 2082–2089, 2009.

[Huang *et al.*, 2007a] C. Huang, H. Ai, Y. Li, and S. Lao. High-performance rotation invariant multiview face detection. *PAMI*, 29(4):671–686, 2007.

[Huang *et al.*, 2007b] G.B. Huang, V. Jain, and E. Learned-Miller. Unsupervised joint alignment of complex images. In *IEEE Intl. Conf. on Computer Vision (ICCV)*, 2007.

[Huang *et al.*, 2007c] G.B. Huang, M Ramesh, T.L. Berg, and Erik Learned-Miller. Labeled Faces in the Wild: A database for studying face recognition in unconstrained environments. *UMass Amherst Technical Report 07-49*, October 2007.

[Huang *et al.*, 2008] G.B. Huang, M.J. Jones, and E. Learned-Miller. LFW results using a combined Nowak plus MERL recognizer. In *Real-Life Images workshop at ECCV*, 2008.

[Hwang *et al.*, 2011] S. J. Hwang, F. Sha, , and K. Grauman. Sharing features between objects and their attributes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Jebara, 2004] T. Jebara. Multi-task feature and kernel selection for SVMs. In *ICML*, 2004.

[Jiang *et al.*, 2008a] Wei Jiang, Shih-Fu Chang, Tony Jebara, and Alexander Loui. Semantic concept classification by joint semi-supervised learning of feature subspaces and support vector machines. In *European Conference on Computer Vision*, pages 270–283, 2008.

[Jiang *et al.*, 2008b] Wei Jiang, Eric Zavesky, Shih-Fu Chang, and Alex Loui. Cross-Domain Learning Methods for High-Level Visual Concept Classification. In *IEEE International Conference on Image Processing*, San Diego, California, U.S.A, October 2008.

[Jiang *et al.*, 2009a] Wei Jiang, Courtenay Cotton, Shih-Fu Chang, Dan Ellis, and Alexander C. Loui. Short-term audio-visual atoms for generic video concept classification. In *Proceeding of ACM international conference on Multimedia (ACM MM)*, October 2009.

[Jiang *et al.*, 2009b] Yu-Gang Jiang, Chong-Wah Ngo, and Shih-Fu Chang. Semantic context transfer across heterogeneous sources for domain adaptive video search. In *Proceeding of ACM international conference on Multimedia (ACM MM)*, October 2009.

[Jin *et al.*, 2008] R. Jin, H. Valizadegan, and H Li. Ranking refinement and its application to information retrieval. In *WWW*, 2008.

[Kennedy and Chang, 2010] Lyndon Kennedy and Shih-Fu Chang. Visual ontology construction and concept detection for multimedia indexing and retrieval. In Phillip C.-Y. Sheu, Heather Yu, C. V. Ramamoorthy, Arvind K. Joshi, and A. Zadeh, editors, *Semantic Computing*. IEEE and John Wiley and Sons, 2010.

[Kirby and Sirovich, 1990] M. Kirby and L. Sirovich. Application of the karhunen-loeve procedure for the characterization of human faces. *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, 12(1):103 –108, jan 1990.

[Koenderink and van Doorn, 1992] Jan Koenderink and Andrea van Doorn. Surface shape and curvature scales. *Image and Vision Computing*, 1992.

[Kontschieder *et al.*, 2009] Peter Kontschieder, Michael Donoser, and Horst Bischof. Beyond pairwise shape similarity analysis. *ACCV*, 2009.

[Kulkarni *et al.*, 2011] Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C. Berg, and Tamara L. Berg. Baby talk: Understanding and generating image descriptions. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Kumar *et al.*, 2008] N. Kumar, P. N. Belhumeur, and S. K. Nayar. FaceTracer: A search engine for large collections of images with faces. In *European Conf. on Computer Vision (ECCV)*, 2008.

[Kumar *et al.*, 2009] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *IEEE Intl. Conf. on Computer Vision (ICCV)*, 2009.

[Kundur and Hatzinakos, 1996] D. Kundur and D. Hatzinakos. Blind image deconvolution. *IEEE Signal Processing Magazine*, (3):43–64, 1996.

[Lafferty *et al.*, 2001] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[Lampert *et al.*, 2009] C.H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[Lazebnik and Raginsky, 2009] S. Lazebnik and M. Raginsky. An empirical bayes approach to contextual region classification. In *CVPR*, 2009.

[Lew *et al.*, 2006] Michael S. Lew, Nicu Sebe, Chabane Djeraba, and Ramesh Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2(1):1–19, 2006.

[Li-Jia Li and Fei-Fei, 2010a] Eric P. Xing Li-Jia Li, Hao Su and Li Fei-Fei. Object bank: A high-level image representation for scene classification and semantic feature sparsification. In *Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2010.

[Li-Jia Li and Fei-Fei, 2010b] Yongwhan Lim Li-Jia Li, Hao Su and Li Fei-Fei. Objects as attributes for scene classification. In *European Conference of Computer Vision (ECCV), International Workshop on Parts and Attributes*, Crete, Greece, September 2010.

[Ling and Jacobs, 2007] H. Ling and D.W. Jacobs. Shape classification using the inner-distance. *TPAMI*, 29:286–299, 2007.

[Ling *et al.*, 2007] H. Ling, S. Soatto, N. Ramanathan, and D.W. Jacobs. A study of face recognition as people age. In *IEEE Intl. Conf. on Computer Vision (ICCV)*, 2007.

[Ling *et al.*, 2010] Haibin Ling, Xingwei Yang, and Longin Latecki. Balancing deformability and discriminability for shape matching. *ECCV*, 2010.

[Liu *et al.*, 2001] Zicheng Liu, Ying Shan, and Zhengyou Zhang. Expressive expression mapping with ratio images. In *SIGGRAPH 01: Proc. of the 28th CGIT*, pages 271–276, 2001.

[Liu, 2011] Jingen Liu. Recognizing human actions by attributes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Lowe, 2003] David Lowe. Distinctive image features from scale-invariant keypoints. *Intl. Journal of Computer Vision (IJCV)*, 2003.

[Maji *et al.*, 2011] Subhransu Maji, Lubomir Bourdev, and Jitendra Malik. Action recognition from a distributed representation of pose and appearance. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Malik, 2003] Shahzad Malik. Digital face replacement in photographs. http://www.cs.toronto.edu/∼smalik/2530/project/results.html, 2003.

[Manay *et al.*, 2006] Siddharth Manay, Daniel Cremers, Byung-Woo Hong, Anthony J. Yezzi, and Stefano Soatto. Integral invariants for shape matching. *TPAMI*, 28:1602–1618, 2006.

[Moghaddam and Yang, 2002] Baback Moghaddam and Ming-Hsuan Yang. Learning gender with support faces. *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, 24(5):707–711, 2002.

[Mokhtarian *et al.*, 1996] Farzin Mokhtarian, Sadegh Abbasi, and Josef Kittler. Efficient and robust retrieval by shape content through curvature scale space. *Workshop on Image Databases and Mult. Search*, 1996.

[Naphade and Smith, 2004] Milind R. Naphade and John R. Smith. On the detection of semantic concepts at trecvid. In *Proceedings of the 12th annual ACM international conference on Multimedia*, MULTIMEDIA '04, pages 660–667, New York, NY, USA, 2004. ACM.

[Naphade *et al.*, 2006] M. Naphade, J.R. Smith, J. Tesic, Shih-Fu Chang, W. Hsu, L. Kennedy, A. Hauptmann, and J. Curtis. Large-scale concept ontology for multimedia. *Multimedia, IEEE*, 13(3):86 –91, july-sept. 2006.

[Newton *et al.*, 2005] Elaine Newton, Latanya Sweeney, and Bradley Malin. Preserving Privacy by De-Identifying Face Images. *IEEE Trans. on Knowledge and Data Eng.*, pages 232–243, 2005.

[Nguyen and Bai, 2010] Hieu V. Nguyen and Li Bai. Cosine similarity metric learning for face verification. In *ACCV*, pages 709–720, 2010.

[Nister and Stewenius, 2006] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2161–2168, 2006.

[Nowak and Jurie, 2007] E. Nowak and F. Jurie. Learning visual similarity measures for comparing never seen objects. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[Nowak *et al.*, 2006] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *European Conf. on Computer Vision (ECCV)*, pages 490–503, 2006.

[Oliva and Torralba, 2001] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42:145–175, 2001.

[Omron, 2010] Omron. OKAO vision. `http://www.omron.com/r_d/coretech/vision/okao.html`, 2010.

[O'Toole *et al.*, 2007] A.J. O'Toole, P.J. Phillips, Fang Jiang, J. Ayyad, N. Penard, and H. Abdi. Face recognition algorithms surpass humans matching faces over changes in illumination. *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, 29(9):1642–1646, September 2007.

[Palatucci *et al.*, 2009] M. Palatucci, D. Pomerleau, G. Hinton, and T. Mitchell. Zero-shot learning with semantic output codes. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

[Parikh and Grauman, 2011] Devi Parikh and Kristen Grauman. Interactively building a discriminiative vocabulary of nameable attributes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Pentland *et al.*, 1994] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 1994.

[Pentland *et al.*, 1996] Alex Pentland, Rosalind Picard, and Stan Sclaroff. Photobook: Content-based manipulation of image databases. *Intl. Journal of Computer Vision (IJCV)*, pages 233–254, 1996.

[Pérez *et al.*, 2003] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, volume 22, pages 313–318, July 2003.

[Perona, 2010] Pietro Perona. Vision of a visipedia. *Proceedings of the IEEE*, 98(8):1526 –1534, Aug. 2010.

[Phillips *et al.*, 2000] P. Phillips, Hyeonjoon Moon, Syed Rizvi, and Patrick Rauss. The FERET evaluation methodology for face-recognition algorithms. *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, 22(10):1090–1104, 2000.

[Phillips *et al.*, 2005] P. Jonathon Phillips, Patrick J. Flynn, Todd Scruggs, Kevin W. Bowyer, Jin Chang, Kevin Hoffman, Joe Marques, Jaesik Min, and William Worek. Overview of the face recognition grand challenge. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[Phillips *et al.*, 2006] P.J. Phillips, P.J. Flynn, T. Scruggs, K.W. Bowyer, and W. Worek. Preliminary face recognition grand challenge results. In *IEEE Conf. on Automatic Face and Gesture Recognition*, 2006.

[Pinto *et al.*, 2009] N. Pinto, J. J. DiCarlo, and D. D. Cox. How far can you get with a modern face recognition test set using only simple features? In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[Pottmann *et al.*, 2009] Helmut Pottmann, Johannes Wallner, Qi-Xing Huang, and Yong-Liang Yang. Integral invariants for robust geometry processing. *Computer Aided Geometric Design*, 26(1):37 – 60, 2009.

[Qin *et al.*, 2008] T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, W.-Y. Xiong, and H. Li. Learning to rank relational objects and its application to web search. In *WWW*, 2008.

[Ramamoorthi and Hanrahan, 2001] Ravi Ramamoorthi and Pat Hanrahan. An Efficient Representation for Irradiance Environment Maps. In *SIGGRAPH 01*, pages 497–500, 2001.

[Rubner *et al.*, 2000] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. The earth mover's distance as a metric for image retrieval. *IJCV*, pages 99–121, 2000.

[Rui *et al.*, 1999] Yong Rui, Thomas S. Huang, and Shih-Fu Chang. Image retrieval: Current techniques, promising directions, and open issues. *Journal of Visual Communication and Image Representation*, 10(1):39 – 62, 1999.

[Russakovsky and Fei-Fei, 2010] Olga Russakovsky and Li Fei-Fei. Attribute learning in large-scale datasets. In *European Conference of Computer Vision (ECCV), International Workshop on Parts and Attributes*, Crete, Greece, September 2010.

[Russell *et al.*, 2008] B.C. Russell, A. Torralba, and K. Murphy. LabelMe: a database and web-based tool for image annotation. *Intl. Journal of Computer Vision (IJCV)*, 77(1-3):157–173, 2008.

[Sadhegi and Farhadi, 2011] A. Sadhegi and A. Farhadi. Recognition using visual phrases. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Samaria and Harter, 1994] F.S. Samaria and A.C. Harter. Parameterisation of a stochastic model for human face identification. In *Workshop on Applications of Computer Vision*, 1994.

[Scheirer *et al.*, 2010] W. Scheirer, A. Rocha, R. Micheals, and T. Boult. Robust fusion: Extreme value theory for recognition score normalization. In *European Conf. on Computer Vision (ECCV)*, 2010.

[Shakhnarovich *et al.*, 2002] G. Shakhnarovich, P.A. Viola, and B. Moghaddam. A unified learning framework for real time face detection and classification. In *IEEE Conf. on Automatic Face and Gesture Recognition*, 2002.

[Siddiquie *et al.*, 2011] Behjat Siddiquie, Rogerio Feris, and Larry Davis. Image ranking and retrieval based on multi-attribute queries. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Sigal *et al.*, 2010] Leonid Sigal, David Fleet, Niko Troje, and Micha Livne. Human attributes from 3d pose tracking. In *European Conf. on Computer Vision (ECCV)*, 2010.

[Sim *et al.*, 2002] T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination, and expression (PIE) database. In *IEEE Conf. on Automatic Face and Gesture Recognition*, pages 46–51, 2002.

[Sinha and Poggio, 1996] P. Sinha and T. Poggio. I think i know that face... *Nature*, 384(6608):404, 1996.

[Sinha *et al.*, 2006] P. Sinha, B. Balas, Y. Ostrovsky, and R. Russell. Face recognition by humans: Nineteen results all computer vision researchers should know about. *Proceedings of the IEEE*, 94:1948–1962, 2006.

[Sivic and Zisserman, 2003] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE Intl. Conf. on Computer Vision (ICCV)*, volume 2, pages 1470–1477, October 2003.

[Smith and Chang, 1996] John R. Smith and Shih-Fu Chang. Visualseek: a fully automated content-based image query system. In *Proceedings of the fourth ACM international conference on Multimedia*, MULTIMEDIA '96, pages 87–98, New York, NY, USA, 1996. ACM.

[Snoek *et al.*, 2005] C. G. M. Snoek, J. C. van Gemert, J. M. Geusebroek, B. Huurnink, D. C. Koelma, G. P. Nguyen, O. de Rooij, F. J. Seinstra, A. W. M. Smeulders, C. J. Veenman, and M. Worring. The mediamill trecvid 2005 semantic video search engine. In *Proceedings of the TRECVID Workshop*, 2005.

[Snoek *et al.*, 2007] C.G.M. Snoek, B. Huurnink, L. Hollink, M. de Rijke, G. Schreiber, and M. Worring. Adding semantics to detectors for video retrieval. *Multimedia, IEEE Transactions on*, 9(5):975 –986, aug. 2007.

[Taigman *et al.*, 2009] Y. Taigman, L. Wolf, and T. Hassner. Multiple one-shots for utilizing class label information. In *British Machine Vision Conference (BMVC)*, 2009.

[Taskar *et al.*, 2003] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *NIPS*, 2003.

[Torresani *et al.*, 2010] Lorenzo Torresani, Martin Szummer, and Andrew Fitzgibbon. Efficient object category recognition using classemes. In *European Conf. on Computer Vision (ECCV)*, 2010.

[Turk and Pentland, 1991] M. Turk and A. Pentland. Face recognition using eigenfaces. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 1991.

[Viola and Jones, 2001] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[Wang and Forsyth, 2009] Gang Wang and David Forsyth. Joint learning of visual attributes, object classes and visual saliency. In *IEEE Intl. Conf. on Computer Vision (ICCV)*, 2009.

[Wang and Mori, 2010] Yang Wang and Greg Mori. A discriminative latent model of object classes and attributes. In *European Conf. on Computer Vision (ECCV)*, 2010.

[Wang *et al.*, 2007a] Hongcheng Wang, Ramesh Raskar, Ning Xu, and Narendra Ahuja. Videoshop: A New Framework for Video Editing in Gradient Domain. *Graphical Models*, 69:57–70, 2007.

[Wang *et al.*, 2007b] Yang Wang, Zicheng Liu, Gang Hua, Zhen Wen, Zhengyou Zhang, and Dimitris Samaras. Face re-lighting from a single image under harsh lighting conditions,. *CVPR '07*, 2007.

[Weissman *et al.*, 2005] T. Weissman, E. Ordentlich, G. Seroussi, S. Verdu, and M.J. Weinberger. Universal discrete denoising: known channel. *Tr. Info. Theory*, 51:5–28, 2005.

[Welinder *et al.*, 2010] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. Technical Report CNS-TR-201, Caltech, 2010.

[Wen *et al.*, 2003] Z. Wen, Z. Liu, and T. S. Huang. Face Relighting with Radiance Environment Maps. In *CVPR '03*, pages 158–165, 2003.

[Wiskott *et al.*, 1997] Laurenz Wiskott, Jean-Marc Fellous, Norbert Krüger, and Christoph von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, 19:775–779, 1997.

[Wolf *et al.*, 2008] L. Wolf, T. Hassner, and Y. Taigman. Descriptor based methods in the wild. In *Real-Life Images Workshop at ECCV*, 2008.

[Wolf *et al.*, 2009] L. Wolf, T. Hassner, and Y. Taigman. Similarity scores based on background samples. In *Asian Conf. on Computer Vision (ACCV)*, 2009.

[Yanagawa *et al.*, 2007] Akira Yanagawa, Shih-Fu Chang, Lyndon Kennedy, and Winston Hsu. Columbia university's baseline detectors for 374 lscom semantic visual concepts. Technical report, Columbia University, March 2007.

[Yang *et al.*, 2005] J. Yang, A.F. Frangi, J. Yang, D. Zhang, and Z. Jin. KPCA plus LDA: A complete kernel fisher discriminant framework for feature extraction and recognition. *PAMI*, 27(2):230–244, 2005.

[Yang *et al.*, 2008] Xingwei Yang, Xiang Bai, Longin Latecki, and Zhuowen Tu. Improving shape retrieval by learning graph transduction. *ECCV*, 2008.

[Yang *et al.*, 2009] Xingwei Yang, S. K.-Tezel, and L.J. Latecki. Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. *CVPR*, 2009.

[yong Neo *et al.*, 2006] Shi yong Neo, Jin Zhao, Min yen Kan, and Tat seng Chua. Video retrieval using high level features: Exploiting query matching and confidence-based weighting. In *CIVR 2006*, pages 143–152. Springer-Verlag, 2006.

[Yu and Aloimonos, 2010] Xiaodong Yu and Yiannis Aloimonos. Attribute-based transfer learning for object categorization with zero or one training example. In *European Conf. on Computer Vision (ECCV)*, 2010.

[Zhao *et al.*, 2003] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35(4):399–458, 2003.

[Zheng *et al.*, 2008] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. In *NIPS*, 2008.