FaceTracer: A Search Engine for Large Collections of Images with Faces

Neeraj Kumar[1,⋆], Peter Belhumeur[1], and Shree Nayar[1]

Columbia University

Abstract. We have created the first image search engine based entirely on faces. Using simple text queries such as "smiling men with blond hair and mustaches," users can search through over 3.1 million faces which have been automatically labeled on the basis of several facial attributes. Faces in our database have been extracted and aligned from images downloaded from the internet using a commercial face detector, and the number of images and attributes continues to grow daily. Our classification approach uses a novel combination of Support Vector Machines and Adaboost which exploits the strong structure of faces to select and train on the optimal set of features for each attribute. We show state-of-the-art classification results compared to previous works, and demonstrate the power of our architecture through a functional, large-scale face search engine. Our framework is fully automatic, easy to scale, and computes all labels off-line, leading to fast on-line search performance. In addition, we describe how our system can be used for a number of applications, including law enforcement, social networks, and personal photo management. Our search engine will soon be made publicly available.

1 Introduction

We have created the first *face* search engine, allowing users to search through large collections of images which have been automatically labeled based on the appearance of the faces within them. Our system lets users search on the basis of a variety of facial attributes using natural language queries such as, "men with mustaches," or "young blonde women," or even, "indoor photos of smiling children." This face search engine can be directed at all images on the internet, tailored toward specific image collections such as those used by law enforcement or online social networks, or even focused on personal photo libraries.

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

⋆ Supported by the National Defense Science & Engineering Graduate Fellowship.

D. Forsyth, P. Torr, and A. Zisserman (Eds.): ECCV 2008, Part IV, LNCS 5305, pp. 340–353, 2008.

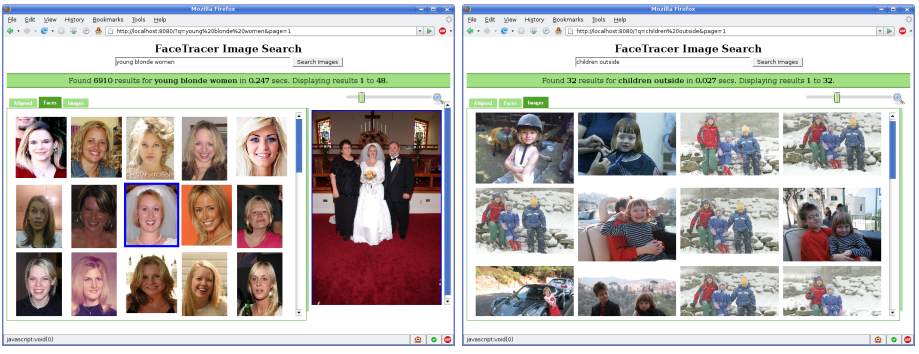
(a) **(b)**

Fig. 1. Results for the query "smiling asian men with glasses," using (a) the Google image search engine and (b) our face search engine. Our system currently has over 3.1 million faces, automatically detected and extracted from images downloaded from the internet, using a commercial face detector [1]. Rather than use text annotations to find images, our system has automatically labeled a large number of different facial attributes on each face (off-line), and searches are performed using only these labels. Thus, search results are returned almost instantaneously. The results also contain links pointing back to the original source image and associated webpage.

simply not annotated. Figures 1a and 1b show the results of the query, "smiling asian men with glasses," using a conventional image search engine (Google Image Search) and our search engine, respectively. The difference in quality of search results is clearly visible. Google's reliance on text annotations results in it finding images that have no relevance to the query, while our system returns only the images that match the query.

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.

Figures 2a and 2b show search results of the queries, "young blonde women" and "children outdoors," respectively. The first shows a view of our extended interface, which displays a preview of the original image in the right pane when the user holds the mouse over a face thumbnail. The latter shows an example of a query run on a personalized set of images. 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.



(a) **(b)**

Fig. 2. Results of queries (a)"young blonde women" and (b) "children outside," using our face search engine. In (a), search results are shown in the left panel, while the right panel shows a preview of the original image for the selected face. (b) shows search results on a personalized dataset, displaying the results as thumbnails of the original images. Note that these results were correctly classified as being "outside" using only the cropped face images, showing that face images often contain enough information to describe properties of the image which are not directly related to faces.

Our search engine owes its superior performance to the following factors:

- **A large and diverse dataset of face images with a significant subset containing attribute labels.** We currently have over 3.1 million aligned faces in our database – the largest such collection in the world. In addition to its size, our database is also noteworthy for being a completely "real-world" dataset. The images are downloaded from the internet and encompass a wide range of pose, illumination, imaging conditions, and were taken using a large variety of cameras. The faces have been automatically extracted and aligned using a commercial face and fiducial point detector [1]. In addition, 10 attributes have been manually labeled on more than 17,000 of the face images, creating a large dataset for training and testing classification algorithms.
- **A scalable and fully automatic architecture for attribute classification.** We present a novel approach tailored toward face classification problems, which uses a boosted set of Support Vector Machines (SVMs) [2] to form a strong classifier with high accuracy. We describe the results of this algorithm on a variety of different attributes, including demographic information such as gender, age, and race; facial characteristics such as eye wear and facial hair; image properties such as blurriness and lighting conditions; and many others as well. A key aspect of this work is that classifiers for new attributes can be trained automatically, requiring only a set of labeled examples. Yet, the flexibility of our framework does not come at the cost of reduced accuracy – we compare against several state-of-the-art classification methods and show the superior classification rates produced by our system.

We will soon be releasing our search engine for public use.

2 Related Work

Our work lies at the intersection of several fields, including computer vision, machine learning, and content-based image retrieval. We present an overview of the relevant work, organized by topic.

Attribute Classification. Prior works on attribute classification have focused mostly on gender and ethnicity classification. Early works such as [3]used neural networks to perform gender classification on small datasets. The Fisherfaces work of [4] showed that linear discriminant analysis could be used for simple attribute classification such as glasses/no glasses. More recently, Moghaddam and Yang [5] used Support Vector Machines (SVMs) [2] trained on small "face-prints" to classify the gender of a face, showing good results on the FERET face database [6]. The works of Shakhnarovich et al. [7] and Baluja & Rowley [8] used Adaboost [9] to select a linear combination of weak classifiers, allowing for almost real-time classification of faces, 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 [10], while the latter uses simple pixel comparison operators.

In contrast, we develop a method that combines the advantages of SVMs and Adaboost (described in Sect. 4). We also present results of an extensive comparison against all three of these prior methods in Sect. 5. Finally, we note that this is an active area of research, and there are many other works on attribute classification which use different combinations of learning techniques, features, and problem formulations [11,12]. An exploration of the advantages and disadvantages of each is beyond the scope of this paper.

Content-Based Image Retrieval (CBIR). Our work can also be viewed as a form of CBIR, where our content is limited to images with faces. Interested readers can refer to the work of Datta et al. [13] for a recent survey of this field. Most relevant to our work is the "Photobook" system [14], 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.

3 Creating the Face Database

To date, we have built a large database of over 3.1 million face images extracted from over 6.2 million images collected from the internet. This database continues to grow as we automatically collect, align, and assign attributes to face images daily. An overview of the database creation process is illustrated in Fig. 3. We download images using two different methods – keyword searches and random downloads. The first allows us to build datasets related to particular terms

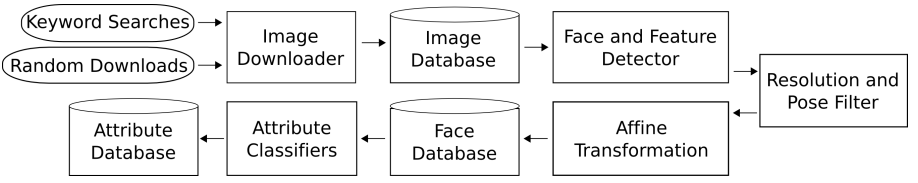


Fig. 3. Overview of database creation. See text for details.

(e.g., celebrity names and professions). The latter allows us to sample from the more general distribution of images on the internet. In particular, it lets us include images that have no corresponding textual information, i.e., that are effectively invisible to current image search engines. Our images are downloaded from a wide variety of online sources, such as Google Images, Microsoft Live Image Search, and Flickr, to name a few. Relevant metadata such as image and page URLs are stored in the EXIF tags of the downloaded images.

Next, we apply the OKAO face detector [1] to the downloaded images to extract faces. This detector also gives us the pose angles of each face, as well as the locations of six fiducial points (the corners of both eyes and the corners of the mouth). We filter the set of faces by resolution and face pose ($\pm 10^\circ$ from front-center). Finally, the remaining faces are aligned to a canonical pose by applying an affine transformation. This transform is computed using linear least squares on the detected fiducial points and corresponding points defined on a template face. (In future work, we intend to go beyond near frontal poses.)

We present various statistics of our current face database in Table 1, divided by image source. We would like to draw attention to three observations about our data. First, from the statistics of randomly downloaded images, it appears that a significant fraction of them contain faces (25.7%), and on average, each image contains 0.5 faces. 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 this respect, our database is similar to the LFW dataset [15], although ours is larger by 2 orders of magnitude and not targeted specifically for face recognition. In contrast, existing face datasets such as Yale Face A&B [16], CMU PIE [17], and FERET [6] are either much smaller in size and/or taken in highly controlled settings. Even the more expansive FRGC version 2.0 dataset [18] has a limited number of subjects, image acquisition locations, and all images were taken with the same camera type. Finally, we have labeled a significant number of these images for our 10 attributes, enumerated in Table 2. In total, we have over 17,000 attribute labels.

4 Automatic Attribute Classification for Face Images

Our approach to image search relies on labeling each image with a variety of attributes. For a dataset as large as ours, it is infeasible to manually label every

Table 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 [1]. Notice that more than 45% of the downloaded images contain faces, and on average, there is one face per two images.

Image Source	# Images Downloaded	# 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 2. List of labeled attributes. The labeled face images are used for training our classifiers, allowing for automatic classification of the remaining faces in our database. Note that these were labeled by a large set of people, and thus the labels reflect a group consensus about each attribute rather than a single user's strict definition.

Attribute/ Options	Number Labeled	Attribute/ Options	Number Labeled	Attribute/ Options	Number Labeled
Gender	**1,954**	**Smiling**	**1,571**	**Race**	**1,309**
Male	867	True	832	White	433
Female	1,087	False	739	Black	399
Age	**3,301**	**Mustache**	**1,947**	Asian	477
Baby	577	True	618	**Eye Wear**	**2,360**
Child	636	False	1,329	None	1,256
Youth	784	**Blurry**	**1,763**	Eyeglasses	665
Middle Aged	815	True	763	Sunglasses	439
Senior	489	False	1,000	**Environment**	**1,583**
Hair Color	**1,033**	**Lighting**	**633**	Outdoor	780
Black	717	Flash	421	Indoor	803
Blond	316	Harsh	212	**Total**	**17,454**

image. Instead, we use our large sets of manually-labeled images to build accurate classifiers for each of the desired attributes.

In creating a classifier for a particular attribute, we could simply choose all pixels on the face, and let our 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 create a rich set of local feature options from which our classifier can automatically select the best ones. Each option consists 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.

Face Regions. We break up the face into a number of functional regions, such as the nose, mouth, etc., much like those defined in the work on modular eigenspaces

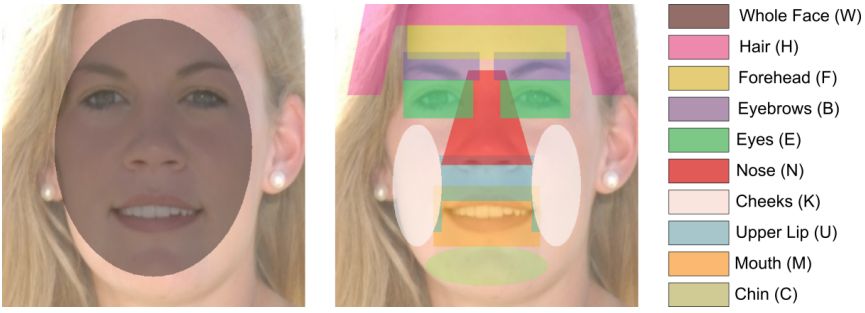


Fig. 4. The face regions used for automatic feature selection. On the left is one region corresponding to the whole face, and on the right are the remaining regions, each corresponding to functional parts of the face. The regions are large enough to be robust against small differences between individual faces and overlap slightly so that small errors in alignment do not cause a feature to go outside of its region. The letters in parentheses denote the code letter for the region, used later in the paper.

[19]. The complete set of 10 regions we use are shown in Fig. 4. 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.

Types of Pixel Data. We include different color spaces and image derivatives as possible feature types. These can often be more discriminative than standard RGB values for certain attributes. Table 3 lists the various options.

Normalizations. Normalizations are important for removing lighting effects, allowing for better generalization across images. We can remove illumination gains by using mean normalization, $\hat{x} = \frac{x}{\mu}$, or both gains and offsets by using energy normalization, $\hat{x} = \frac{x-\mu}{\sigma}$. 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.

Aggregations. For some attributes, aggregate information over the entire region might be more useful than individual values at each pixel. This includes histograms of values over the region, or simply the mean and variance.

To concisely refer to a complete feature option, we define a shorthand notation using the format, "Region:pixel type.normalization.aggregation." The region notation is shown in Fig. 4; the notation for the pixel type, normalization, and aggregation is shown in Table 3.

4.1 Classifier Architecture

In recent years, Support Vector Machines (SVMs) [2] have been used successfully for many classification tasks [20,21]. SVMs aim to find the linear hyperplane which best separates feature vectors of two different classes, so as to

Table 3. Feature type options. A complete feature type is constructed by first converting the pixels in a given region 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. The letters in parentheses are used as code letters in a shorthand notation for concisely designating feature types.

Pixel Value Types	Normalizations	Aggregation
RGB (r)	None (n)	None (n)
HSV (h)	Mean-Normalization (m)	Histogram (h)
Image Intensity (i)	Energy-Normalization (e)	Statistics (s)
Edge Magnitude (m)		
Edge Orientation (o)		

simultaneously minimize the number of misclassified examples (training error) and maximize the distance between the classes (the *margin*).

As with many classification algorithms, SVMs perform best when given only the relevant data – too many extraneous inputs can confuse or overtrain the classifier, resulting in poor accuracy on real data. In particular, if we would like to train a classifier for an attribute that is only dependent on a certain part of the face (e.g., "is smiling?"), giving the SVM a feature vector constructed from all the pixels of the face is unlikely to yield optimal results. Given the large number of regions and feature types described in the previous section, an efficient and automatic selection algorithm is needed to find the optimal combination of features for each attribute. Following the successes of [10,7,8,11], we use Adaboost [9] for this purpose.

Adaboost is a principled, iterative approach for building strong classifiers out of a collection of "weak" classifiers. In each iteration of Adaboost, the weak classifier that best classifies a set of weighted examples is greedily picked to form part of the final classifier. The weights on the examples are then adjusted to make misclassified examples more important in future iterations, and the process is repeated until a given number of weak classifiers has been picked. A major advantage of Adaboost is that it is resistant to overtraining [22,23].

We combine the strengths of these two methods by constructing a number of "local" SVMs and letting Adaboost create an optimal classifier using a linear combination of them. We create one SVM for each region, feature type, and SVM parameter combination, using the LibSVM library [24]. Normally, Adaboost is performed using weak classifiers, which need to be retrained at the beginning of each round. However, we rely on the fact that our local SVMs will either be quite powerful (if created using the relevant features for the current attribute), or virtually useless (if created from irrelevant features). Retraining will not significantly improve the classifiers in either case.

Accordingly, we precompute the results of each SVM on all examples, one SVM at a time. Thus, our classifiers remain fixed throughout the Adaboost process, and we do not need to keep a large number of SVMs in memory. Once all SVM outputs have been computed, we run our Adaboost rounds to obtain the

Table 4. Error rates and top feature combinations for each attribute, computed by training on 80% of the labeled data and testing on the remaining 20%, averaging over 5 runs (5-fold cross-validation). Note that the attribute-tuned global SVM performs as well as, or better than, the local SVMs in all cases, and requires much less memory and computation than the latter. The top feature combinations selected by our algorithm are shown in ranked order from more important to less as "Region:feature type" pairs, where the region and feature types are listed using the code letters from Fig. 4 and Table 3. For example, the first combination for the hair color classifier, "H:r.n.s," takes from the hair region (H) the RGB values (r) with no normalization (n) and using only the statistics (s) of these values.

Attribute	**Error Rates for Attribute-Tuned Local SVMs**	**Error Rates for Attribute-Tuned Global SVM**	**Top Feature Combinations in Ranked Order** Each combination is represented as Region:pixtype.norm.aggreg
Gender	9.42%	8.62%	W:i.m.n | W:o.n.n | W:i.n.n | W:i.e.n
Age	17.34%	16.65%	W:i.m.n | W:i.n.n | H:r.e.n | E:r.m.n | H:r.e.s | W:o.n.n
Race	7.75%	6.49%	W:i.m.n | E:r.e.n | C:o.n.n | M:r.m.n | W:o.n.n
Hair Color	7.85%	5.54%	H:r.n.s | W:i.m.n | E:r.m.n | H:r.n.n | W:i.n.n | H:r.m.n
Eye Wear	6.22%	5.14%	W:m.n.n | W:i.n.n | K:o.n.h | W:m.m.n | N:r.n.n
Mustache	6.42%	4.61%	U:r.e.n | M:r.m.n
Smiling	4.60%	4.60%	M:r.m.n | M:r.n.n | M:r.e.n | W:i.n.n | W:i.e.n | M:i.n.n
Blurry	3.94%	3.41%	W:m.m.n | H:m.n.n | W:m.n.n | H:m.m.n | M:m.m.n
Lighting	2.82%	1.61%	W:i.n.n | W:i.e.n | K:r.n.n | C:o.n.n | E:o.n.n
Environment	12.25%	12.15%	N:r.m.n | K:r.e.n | K:r.m.n | W:r.m.n | E:r.m.n

weights on each SVM classifier. We use the formulation of Adaboost described in [8], with the modification that errors are computed in a continuous manner (using the confidence values obtained from the SVM classifier), rather than discretely as is done in [8]. We found this change improves the stability of the results, without adversely affecting the error rates.

The error rates of these "attribute-tuned local SVMs" are shown in the second column of Table 4. The rates were computed by dividing the labeled examples for each attribute into 5 parts, using 4 parts to train and the remaining one to test, and then rotating through all 5 sets (5-fold cross-validation). Note that in most cases, our error rates are below 10%, and for many attributes, the error rate is under 5%. (The higher error rates for age are due to the fact that different people's labels for each of the age categories did not match up completely.)

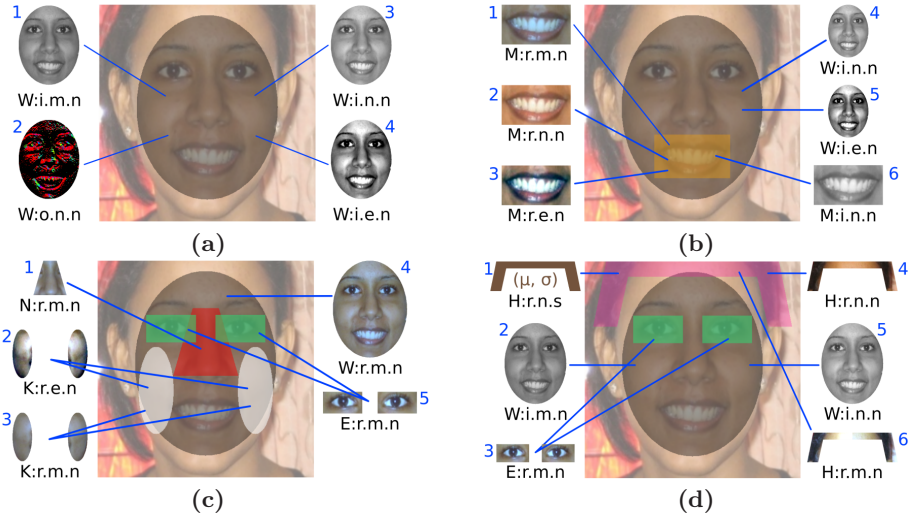


Fig. 5. Illustrations of automatically-selected region and feature types for (a) gender, (b) smiling, (c) environment, and (d) hair color. Each face image is surrounded by depictions of the top-ranked feature combinations for the given attribute, along with their corresponding shorthand label (as used in Table 4). Notice how each classifier uses different regions and feature types of the face.

We emphasize the fact that these numbers are computed using our real-world dataset, and therefore reflect performance on real images.

A limitation of this architecture is that classification will require keeping a possibly large number of SVMs in memory, and each one will need to be evaluated for every input image. Furthermore, one of the drawbacks of the Adaboost formulation is that different classifiers can only be combined linearly. Attributes which might depend on non-linear combinations of different regions or feature types would be difficult to classify using this architecture.

We solve both of these issues simultaneously by training one "global" SVM on the union of the features from the top classifiers selected by Adaboost. We do this by concatenating the features from the N highest-weighted SVMs (from the output of Adaboost), and then training a single SVM classifier over these features (optimizing over N). In practice, the number of features chosen is between 2 (for "mustache") and 6 (e.g., for "hair color"). Error rates for this algorithm, denoted as "Attribute-Tuned Global SVM," are shown in the third column of Table 4. Notice that for each attribute, these rates are equal to, or less than, the rates obtained using the combination of local SVMs, despite the fact that these classifiers run significantly faster and require only a fraction of the memory (often less by an order of magnitude).

The automatically-selected region and feature type combinations for each attribute are shown in the last column of Table 4. Listed in order of decreasing importance, the combinations are displayed in a shorthand notation using the codes given in Fig. 4 and Table 3. In Fig. 5, we visually illustrate the top feature

Table 5. Comparison of classification performance against prior methods. Our attribute-tuned global SVM performs better than prior state-of-the-art methods. Note the complementary performances of both Adaboost methods versus the full-face SVM method for the different attributes, showing the strengths and weaknesses of each method. By exploiting the advantages of each method, our approach achieves the best performance.

Classification Method	Gender Error Rate	Smiling Error Rate
Attribute-Tuned Global SVM	**8.62%**	**4.60%**
Adaboost (pixel comparison feats.) [9]	13.13%	7.41%
Adaboost (Haar-like feats.) [8]	12.88%	6.40%
Full-face SVM [6]	9.52%	13.54%

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.

5 Comparison to Prior Work

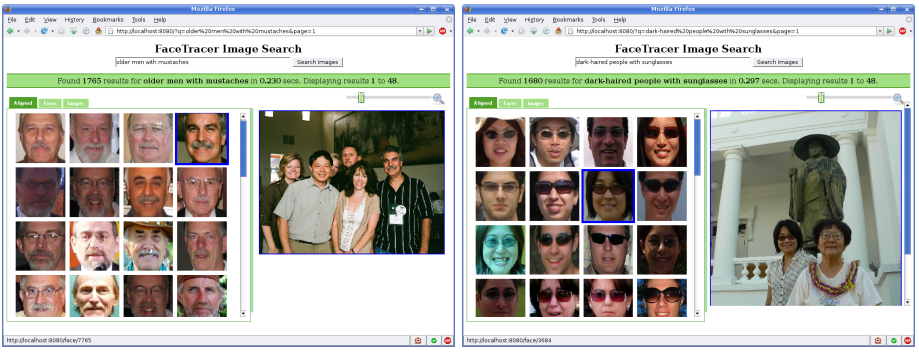
While we have designed our classifier architecture to be flexible enough to handle a large variety of attributes, it is important to ensure that we have not sacrificed accuracy in the process. We therefore compare our approach to three state-of-the-art methods for attribute classification: full-face SVMs using brightness normalized pixel values [5], Adaboost using Haar-like features [7], and Adaboost using pixel comparison features [8]. Since these works have mostly focused on gender classification, we use that attribute as our first testing criteria.

The error rates for gender classification using our training and testing data on all methods are shown in the second column of Table 5. We note that our method performs slightly better than the prior SVM method and significantly better than both Adaboost methods. The difference between the Adaboost and SVM methods may reflect one limitation of using linear combinations of weak classifiers – the classifiers might be too weak to capture all the nuances of gender differences.

To see how these methods do on a localized attribute, we also applied each of them to the "smiling" attribute. Here, while once again our method has the lowest error rate, we see that the Adaboost methods perform significantly better than the prior SVM method. This result highlights the power of Adaboost to correctly find the important features from a large set of possibilities, as well as the degradation in accuracy of SVMs when given too much irrelevant data.

6 The FaceTracer Engine

We have trained attribute-tuned global SVM classifiers for each attribute listed in Table 4. In an offline process, all images in our database are sent through the classifiers for each attribute, and the resulting attribute labels are stored for fast online searches using the FaceTracer engine.



(a) **(b)**

Fig. 6. Results of queries (a) "older men with mustaches" and (b) "dark-haired people with sunglasses" on our face search engine. The results are shown with aligned face images on the left, and a preview of the original image for the currently selected face on the right. Notice the high quality of results in both cases.

For a search engine, the design of the user interface is important for enabling users to easily find what they are looking for. We use simple text-based queries, since these are both familiar and accessible to most internet users. Search queries are mapped onto attribute labels using a dictionary of terms. Users can see the current list of attributes supported by the system on the search page, allowing them to construct their 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 different languages.

Results are ranked in order of decreasing confidence, so that the most relevant images are shown first. (Our classifier gives us confidence values for each labeled attribute.) For searches with multiple query terms, we combine the confidences of different labels such that the final ranking shows images in decreasing order of relevance to all search terms. To prevent high confidences for one attribute from dominating the search results, we convert the confidences into probabilities, and then use the product of the probabilities as the sort criteria. This ensures that the images with high confidences for *all* attributes are shown first.

Example queries on our search engine are shown in Figs. 1b, 2, and 6. The returned results are all highly relevant, and the user can view the results in a variety of ways, as shown in the different examples. Figure 2b 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.

Our search engine can be used in many other applications, replacing or augmenting existing tools. In law enforcement, eyewitnesses to crimes could use our system to quickly narrow a list of possible suspects and then identify the actual criminal from this reduced list, saving time and increasing the chances of finding the right person. On the internet, our face search engine is a perfect match for

social networking websites such as Facebook and Myspace, which contain large numbers of images with people. Additionally, the community aspect of these websites would allow for collaborative creation of new attributes. Finally, users can utilize our system to more easily organize and manage their own personal photo collections. For example, searches for blurry or other poor-quality images can be used to find and remove all such images from the collection.

7 Discussion

In this work, we have described a new approach to searching for images in large databases and have constructed the first face search engine using this approach. By limiting our focus to images with faces, we are able to align the images to a common coordinate system. This allows us to exploit the commonality of facial structures across people to train accurate classifiers for real-world face images. Our approach shows the power of combining the strengths of different algorithms to create a flexible architecture without sacrificing classification accuracy.

As we continue to grow and improve our system, we would also like to address some of our current limitations. For example, to handle more than just frontal faces would require that we define the face regions for each pose bin. Rather than specifying the regions manually, however, we can define them once on a 3D model, and then project the regions to 2D for each pose bin. The other manual portion of our architecture is the labeling of example images for training classifiers. Here, we can take advantage of communities on the internet by offering a simple interface for both defining new attributes and labeling example images. Finally, while our dictionary-based search interface is adequate for most simple queries, taking advantage of methods in statistical natural language processing (NLP) could allow our system to map more complex queries to the list of attributes.

Acknowledgements. We are grateful to Omron Technologies for providing us the OKAO face detection system. This work was supported by NSF grants IIS-03-08185 and ITR-03-25867.

References

1. Omron: OKAO vision (2008), `http://www.omron.com/rd/vision/01.html`
2. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3) (1995)
3. Golomb, B.A., Lawrence, D.T., Sejnowski, T.J.: Sexnet: A neural network identifies sex from human faces. NIPS, 572–577 (1990)
4. Belhumeur, P.N., Hespanha, J., Kriegman, D.J.: Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. In: Buxton, B.F., Cipolla, R. (eds.) ECCV 1996. LNCS, vol. 1065, pp. 45–58. Springer, Heidelberg (1996)
5. Moghaddam, B., Yang, M.-H.: Learning gender with support faces. TPAMI 24(5), 707–711 (2002)
6. Phillips, P., Moon, H., Rizvi, S., Rauss, P.: The FERET evaluation methodology for face-recognition algorithms. TPAMI 22(10), 1090–1104 (2000)

7. Shakhnarovich, G., Viola, P.A., Moghaddam, B.: A unified learning framework for real time face detection and classification. ICAFGR, 14–21 (2002)
8. Baluja, S., Rowley, H.: Boosting sex identification performance. IJCV (2007)
9. Freund, Y., Shapire, R.E.: Experiments with a new boosting algorithm. In: ICML (1996)
10. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR (2001)
11. Bartlett, M.S., Littlewort, G., Fasel, I., Movellan, J.R.: Real time face detection and facial expression recognition: Development and applications to human computer interaction. CVPRW 05 (2003)
12. Wang, Y., Ai, H., Wu, B., Huang, C.: Real time facial expression recognition with adaboost. In: ICPR, pp. 926–929 (2004)
13. Datta, R., Li, J., Wang, J.Z.: Content-based image retrieval: Approaches and trends of the new age. Multimedia Information Retrieval, 253–262 (2005)
14. Pentland, A., Picard, R., Sclaroff, S.: Photobook: Content-based manipulation of image databases. IJCV, 233–254 (1996)
15. Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49 (2007)
16. Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J.: From few to many: Illumination cone models for face recognition under variable lighting and pose. TPAMI 23(6), 643–660 (2001)
17. Sim, T., Baker, S., Bsat, M.: The CMU pose, illumination, and expression (PIE) database. In: ICAFGR, pp. 46–51 (2002)
18. Phillips, P.J., Flynn, P.J., Scruggs, T., Bowyer, K.W., Chang, J., Hoffman, K., Marques, J., Min, J., Worek, W.: Overview of the face recognition grand challenge. CVPR, 947–954 (2005)
19. Pentland, A., Moghaddam, B., Starner, T.: View-based and modular eigenspaces for face recognition. CVPR, 84–91 (1994)
20. Huang, J., Shao, X., Wechsler, H.: Face pose discrimination using support vector machines (SVM). In: ICPR, pp. 154–156 (1998)
21. Osuna, E., Freund, R., Girosi, F.: Training support vector machines: An application to face detection. CVPR (1997)
22. Schapire, R., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods. The Annals of Statistics 26(5), 1651–1686 (1998)
23. Drucker, H., Cortes, C.: Boosting decision trees. NIPS, 479–485 (1995)
24. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), `http://www.csie.ntu.edu.tw/cjlin/libsvm/`