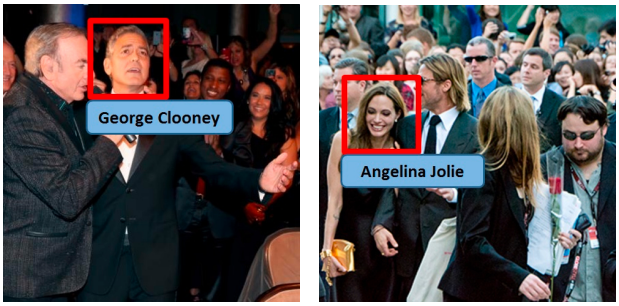
Evaluating Open-Universe Face Identification on the Web

Brian C. Becker[†] and Enrique G. Ortiz[‡]
[†] Robotics Institute, Carnegie Mellon University, Pittsburgh, PA
[‡] Center for Research in Computer Vision, University of Central Florida, Orlando, FL
brian@briancbecker.com and eortiz@cs.ucf.edu

Abstract

Face recognition is becoming a widely used technique to organize and tag photos. Whether searching, viewing, or organizing photos on the web or in personal photo albums, there is a growing demand to index real-world photos by the subjects in them. Even consumer platforms such as Google Picasa, Microsoft Photo Gallery, and social network sites such as Facebook have integrated forms of automated face tagging and recognition; furthermore, a number of libraries and cloud-based APIs that perform face recognition have become available. With such a plethora of choices, comparisons of recent advances become more important to gauge the state of progress in the field. This paper evaluates face identification in the context of not only research algorithms, but also considers consumer photo products, client-side libraries, and cloud-based APIs on a new, large-scale dataset derived from PubFig83 and LFW in a realistic open-universe scenario.



(a) (b)

Figure 1. In many real-world applications, face recognition consists of identifying faces from a set of classes in a training gallery (*e.g.* George Clooney and Angelina Jolie) while ignoring all other background (or distractor) faces. In this case, face identification algorithms must output not only an identity, but a confidence as well; thus, distractor faces can be rejected and only faces belonging to a predefined set of identities recognized.

1. Introduction

Face recognition, a popular topic of research for several decades, has matured significantly in recent years, especially in real-world applications where realistic, uncontrolled faces must be robustly identified. Increasingly better and less expensive consumer hardware, especially cameras, computers, storage, and Internet speeds, have spurred growth in personal photo collections. Furthermore, the amount of video has grown significantly as well, both home videos stored on the computer or uploaded to the Internet, and the availability of movies, security camera videos, and webstreams. As the amount of photo and video content grows, organizing and searching for photos becomes an increasingly difficult task. Face recognition offers a way to tag photos automatically by the people they contain, allowing easier indexing and searching by people of interest.

In fact, recently, face recognition has become mature enough to be increasingly integrated into consumer products. Apple and Google first released face recognition engines built into their respective products iPhoto and Picasa Web in 2008, and were later followed by deployments from Facebook and Microsoft in 2010. New cloud-based APIs, first popularized by face.com (later bought by Facebook), offer very easy to use server-side solutions for monthly subscriptions. Moreover, face recognition is being increasingly integrated into more specialized devices such as smartphones, game consoles, and other gadgets, like the new Android Face Unlock authentification feature.

Currently, there exists several standard benchmarks and datasets (e.g. Ext. Yale B [6], AR [7], and FERET [10]) for evaluating face recognition variations in illumination, pose, expression, etc. However, performance on these datasets has become relatively saturated. In uncontrolled environments, there are comparisons in face verification, where the task is to determine similarity between two face images, and in face identification, where the task is to determine the identity of a face. However, few of these comparisons and datasets consider realistic usage scenarios or compare to a wider range of available face recognition solutions.

This paper introduces a new, open-universe dataset that combines two existing datasets (PubFig83 [12] and LFW [3]) to simulate the far more realistic context for face recognition, where the goal is identifying specific celebrities in crowded environments while rejecting (i.e. not labeling) background faces. The key contribution is the evaluation of not just other research algorithms published in computer vision and biometrics venues [8,9,13,15–18], but an exploration of open-universe face identification performance across multiple segments of the field. Furthermore, unlike vendor tests like the Face Recognition Vendor Test (FRVT) [11], a wide variety of solutions, not just those that voluntarily enroll, are considered.

In this paper, we perform a shallow, but wide preliminary survey of face recognition, using the realistic, web-based challenge of identifying only particular celebrities in web photos as a basis for evaluating the state of the field. Aimed primarily as a benchmarking paper, we target four different segments of face recognition: research algorithms, client-side libraries, cloud-based APIs, and consumer applications. Section 2 begins by providing a short background on face recognition and introduces the dataset we use for real-world face identification in an open-universe scenario. Section 3 provides an overview of the software, methods, and evaluations used in this paper. Section 4 shows results on the introduced dataset across a wide segment of face recognition approaches. As this is only a preliminary study, Section 5 ends with a summary, key-insights, and ideas for future work to more fully evaluate face recognition.

2. Background & Dataset

In this section, we outline three general approaches to face recognition and introduce a new dataset based on the Public Figures [5] and Labeled Faces in the Wild [3] datasets.

2.1. Closed-Universe Face Recognition

Face recognition has long been a popular research topic in academia, starting with the landmark Eigenfaces approach of Turk *et al*. [14]. At the start, faces were drawn from constrained environments where pose, illumination, *etc*. were highly controlled amongst a limited number of people. Most papers assumed a closed-universe, where the sole goal of face recognition was to provide the identity of a test face only drawn from the set of known identities in the training gallery. In this scenario, there is no concept of an impostor or distractor face representing an individual not in the training gallery. However, closed-universe face recognition is of limited use because for most real-world applications, there are many faces of identities that the algorithm encounters that it should ignore. For instance, Pinto *et al*.'s [12] introduction and evaluation of the PubFig83 dataset assumes that all faces presented to the algorithm will have an identity contained in the 83 training gallery individuals. For applications such as finding pictures of celebrities on the web or identifying actors in movies, such an assumption is unreasonable as there will be a great deal of faces of random people in the background.

2.2. Open-Universe Face Verification

In an effort to facilitate the development of face recognition algorithms designed for real-world photos where the pose, illumination, *etc*. is not controlled, Labeled Faces in the Wild (LFW) was introduced in [3]. Going back to the foundations of face recognition, they posed the task as determining if two faces represent the same individual (or identity). This dataset and concept has been a catalyst for the move from limited, artificial datasets to more natural, realistic faces and has resulted in an intense focus on more robust algorithms. Because many of these methods focus on comparing two images and determining if they are the same or not same, some non-straightforward modifications would need to be made to these sets of algorithms to robustly perform face identification, which would most likely result in additional computational overhead.

2.3. Open-Universe Face Identification

In contrast to closed-universe face recognition, which seeks to only provide an identity given a face, and open-universe face verification, which only provides same/not same predictions for pairs of images, open-universe face identification seeks to identify a set of individuals it has been trained on while ignoring faces from all other individuals. For instance, consider the task of finding celebrities and other public figures on the web: many, if not most of the faces encountered on the Internet are not of a particular set of celebrities and should be ignored, i.e. not labeled by the algorithm. In essence, open-universe face identification must perform closed-universe face recognition with one important distinction: in addition to providing a prediction of the identity of the face, the algorithm must also provide a confidence of the prediction. By setting a threshold on this confidence, background faces can be rejected.

2.4. Open-Universe PubFig83+LW Dataset

Pinto *et al*. [12] introduced a new dataset PubFig83 for face identification research that is a modified subset of the Public Figures [5] dataset originally intended for application to face verification. They removed duplicate and near-duplicate photos, cleaned up the database, and removed individuals with few photos. Similar to [9], we introduce a new, open-universe dataset that is suited for manual evaluation as well as automatic evaluation. The new dataset combines the PubFig83 and LFW datasets, where the 83 individuals from PubFig represent the test images and training gallery, and all the remaining individuals from the LFW

dataset represent the distractor gallery or background faces.

To create the PubFig83+LFW dataset, we randomly divided all the faces from each individual in PubFig83 into two thirds training faces and one third testing faces. We then removed any overlapping individuals from LFW and added them as distractors to PubFig83. Face images were resized to 250x250, following LFW conventions and ensuring photos would not be too small for consumer applications to scan for faces. To improve the chances that all the faces would be suitable for a wide variety of face detection and analysis algorithms, we ran all the face images through the SHORE, PittPatt, Google Picasa, Microsoft Photo Gallery, and Apple iPhoto face detectors and only kept faces that were detected by every software package. It is likely there are faces still present that will pose analysis problems for a particular algorithm, but hopefully this chance is reduced.

The resulting PubFig83+LFW dataset has 83 individuals with 8,720 faces for training and 4,282 faces for testing and over 5,000 individuals from LFW with 12,066 faces for background and distractor faces that algorithms should reject. The PubFig83+LFW dataset will be released to the author's website (http://pubfig83lfw.briancbecker.com) in its entirety, including raw images, aligned images, feature representations, and source code for analyzing algorithm performance. A sample subset is shown in Figure 2.

Figure 2. Example face photos from the PubFig83+LFW dataset. The top three rows are sample faces from training individuals from PubFig83: George Clooney, Angelina Jolie, and Ehud Olmert. The bottom row contains distractors from the LFW dataset that serve as background faces that face identification algorithms should reject as not being part of the 83 people to recognize.

3. Methods & Evaluation

In this section, we overview four different segments or domains of face recognition: Research, Client-Side, Consumer, and Cloud-Based, which is a non-exhaustive, but representative list of common face identification platforms. We also describe evaluation metrics so face recognition approaches are reasonably comparable.

3.1. Research Algorithms

A very large number of face recognition algorithms have been described over the years in research papers; in fact, a comprehensive survey would require much more space than is available here in this paper. Hence, we focus on a select subset of popular and recently introduced face identification algorithms. Recently, using linear combinations of the training gallery to represent test faces has gained popularity, thus many of the methods we test with are derivatives of this robust approach. We omit face verification algorithms, such as those popular on the LFW dataset, because taking the classification of pairwise faces as same/not same and applying it to open-universe identification is still an open question not well addressed by current literature.

3.1.1 Face Preprocessing

For a fair comparison between all face identification approaches and to keep classification results comparable, we apply a consistent set of preprocessing techniques to all images. All images are first aligned with a similarity transform using the eye-positions reported by the PittPatt SDK. Images are cropped and illumination-compensated by applying a plane normalization and histogram normalization. Finally, a standard set of features (raw pixels, HOG, Gabor, and LBP) were extracted and concatenated to form a 8,846 length feature vector. Features are unit normalized and reduced to 1,024 dimensions with PCA. Given a face detection with eye positions, the process of aligning, normalizing, and extracting features takes 30-50 ms per face.

3.1.2 Research Methods

For the purposes of this paper, we focus on algorithms of historical importance (nearest neighbor and SVMs) and new linear representation-based algorithms such as SRC and least-squares that have been shown to be robust.

Nearest Neighbor (NN) is perhaps the simplest of all approaches, selecting the identity of a test face by finding the identity of the closest training sample in Euclidian space. Confidence is measured as the distance magnitude.

Support Vector Machines (SVMs) is a binary classification technique that calculates a separating hyperplane between a positive and negative class. We use the fast,

large-scale LIBLINEAR [2] SVM training library to generate one-vs-all models using a linear kernel in high dimensional space. Distance from the best separating hyperplane is used as the confidence.

Sparse Representation-Based Classification (SRC) in [17] presented the principle that a given test image can be represented by a linear combination of images from a large dictionary of faces. The key concept is enforcing sparsity, since a test face can be reconstructed best from a small subset of the dictionary, *i.e*. training faces of the same class. Class residuals or a sparsity concentration index (SCI) can be used as a metric for linear combination algorithms.

Least Squares (L2), following SRC, in [13] claimed that a test face can be represented by a linear combination of the training gallery, however the strict sparsity constraint is unnecessary and finds the coefficient vector by least-squares.

LLC [15], similar to the least-squares method, loosened the sparsity constraint emphasizing that locality is more important than sparsity. Wang *et al*. discovered such a coefficient vector via a constrained (weighted) least-squares approximation with good results on controlled datasets.

K-Nearest Neighbor-SRC (KNN-SRC) in [8] aims to reduce the computational complexity inherent in ℓ^1-minimization by first approximating relevant dictionary elements using K-nearest neighbor to a much smaller dictionary and then running ℓ^1-minimization.

Linearly Approximated SRC (LASRC) in [9] focuses on vastly speeding up the popular SRC face recognition algorithm while retaining high accuracy and robustness. They found that in web-scale, open-universe face identification, the magnitude of the coefficients representing the least-squares solution are highly correlated to the sparse (non-zero) coefficients returned by sparse ℓ^1-minimization solutions. They showed that since zero coefficients have no effect on the minimization process to derive a sparse solution, zero coefficients can be discarded from the minimization if they are known a-priori. Thus, they use least squares to approximate the solution very rapidly to obtain a set of candidate coefficients and then perform a more robust ℓ^1-minimization on only a small fraction of all training faces.

3.2. Client-Side Libraries

A wide variety of client-side face recognition libraries are available for integration into one's own custom software or as an add-on to existing software; in fact, so many exist that evaluating them all is beyond the scope of this paper. Instead, we evaluate two open-source libraries and a commercial software package developed by a company later bought by Google.

OpenCV (version 2.4.2) is a popular and long-standing open-source computer vision library that has recently added a face recognition module (cv::FaceRecognizer). Although it has traditional algorithms such as Eigenfaces and Fisherfaces, we use the more powerful recognition algorithm originally introduced as Circular Local Binary Pattern Histograms (LBPH) [1]. The output of the OpenCV API provides both a best guess for identity and a confidence.

OpenBR (version 0.2.0) is a new open-source biometrics library in beta based on the algorithm described in Klare *et al*. [4]. It performs face verification pair-matching between two faces, i.e. given two faces, it aligns them, extracts features, and provides a similarity score. Because obtaining a similarity score is very fast, each new test face is compared to every training face and the score recorded. We experimented with several methods of using pair-wise similarity scores to determine a best-guess identity and confidence. Unfortunately, OpenBR seems to return many high similarity scores, making a best-match approach infeasible. The method that worked the best for determining identity was to calculate the number of high-match scores per class as a percentage and use that as a confidence as well.

Pittsburgh Pattern Recognition (PitPatt) (last available version) produced a commercial software library SDK before they were bought by Google that provides many capabilities in face detection, analysis, and recognition. Again, they use pair-wise scores, but unlike OpenBR, we found that selecting the highest similarity-score in the training gallery yielded the best results, with performance higher than calculating the mean or median score per class. The highest score was used also used as the confidence. The authors would like to thank Pittsburgh Pattern Recognition for providing their software library for research purposes.

3.3. Cloud-Based APIs

With social media sites such as Facebook adding face recognition to their online service, the demand for web-based face recognition services has grown. Such cloud-based APIs provide the ability to detect faces, associate tags to faces, train, and recognize newly detected faces - with all processing and storage being hosted on a remote server. The company face.com was a predominate leader in the field before being bought by Facebook, but other services have sprung up in its place.

LambdaLabs (www.lambdal.com), one of the first replacements of face.com as a cloud-based API for face recognition, has a limit of 50 identities for training up the face recognizer. Since there are 83 individuals in the PubFig83+LFW dataset, LambdaLabs is too limited in its current beta to be useful in this paper.

ReKognition (www.rekognition.com) is a cloud API that performs face, scene, and other computer vision analysis. Currently, it claims to have three algorithms developed, but they state only the simplest (and presumably fastest) one is currently being exposed to the user.

SkyBiometry (www.skybiometry.com), another a cloud-based API that has a drop-in replacement for the

face.com API, does not have a limit on identities, but does limit to a total of 1,000 total training faces. Since this was a hard limitation of the system, we trained 12 randomly selected faces per identity for a total of 996 training faces.

3.4. Consumer Applications

In the past few years, major vendors of consumer photo gallery applications have integrated face recognition functionality into their software to provide easy and semi-automated ways to tag, organize, and search photos by faces of labeled individuals. To test these applications, we organized the test and training faces into folders per individual and put all of the distractor faces into a single folder for easier importing and labeling. Although there is no programmatic interface to most consumer applications rendering a full evaluation impossible, in the case where the application offered options for changing recognition thresholds/strictness, we hand-evaluated the test sequence several times. To avoid bias, all faces are left raw and unaligned. We consider three popular photo gallery applications and describe how we trained and evaluated them:

Microsoft Photo Gallery (version 2012) is the built-in Windows Live photo gallery software from Microsoft. All training photos were added to the software and batch tagging was performed by selecting a range of faces via the first and last image for each individual and assigning an identity for each of the 83 classes. Evaluation is performed by loading the parent directory of all the identity folders, and for each class filtering both the top level folder of all test faces and the identities folder to get false + true positives and true positives, respectively. We then load in the distractors and filter by each class to obtain the false positives from background faces.

Apple iPhoto (version '11) is a popular OS X photo gallery application. While it includes face recognition software, the user interface is significantly limited compared to Microsoft Photo Gallery and Google Picasa. While the latter allow batch tagging of multiple faces, iPhoto only allows single-face tagging or partial batch tagging based on suggestions it gives. Furthermore, for evaluation, iPhoto appears to lack a way to filter a folder or photo collection by a given individual. Given the limited user interface options for bulk labeling/evaluation, we omit this application.

Google Picasa (version 3.9.0) is a Windows photo organization application that focuses on search. For training, we added all training faces to the application and swapped to the Unnamed Faces section of the app. To label each identity of the 83 classes, we searched for each individual by name, which brought up unnamed faces in the training folder for the individual. A batch name operation for all faces in the individual's folder was performed once per class. For evaluation, we imported the test folders and faces and allowed recognition to run. We then filtered per individual and recorded the number of unnamed faces, number of faces falsely identified, and number of faces correctly identified. We also imported the distractor faces and recorded how many were falsely identified.

Algorithm	Representation	Accuracy (%)
SVM	V1-like-Plus	75.6 ± 0.3
SVM	HT-L3-1st	$\mathbf{87.1 \pm 0.6}$
face.com	Proprietary	82.1 ± 0.5
SVM	HOG+LBP+Gabor	85.9 ± 0.5
LASRC	HOG+LBP+Gabor	83.6 ± 1.3

Table 1. Closed-universe results for various algorithms on the PubFig83 dataset following the experimental setup described in [12].

4. Results

In this section, we describe results across multiple segments of face recognition algorithms in an open-universe scenario of finding public figures while rejecting background figures, using the PubFig83+LFW dataset.

4.1. Closed-Universe Accuracy on PubFig83

We believe that closed-universe accuracy is not a very representative way to measure real-world performance, especially since, in our experience, high closed-universe accuracy is not necessarily indicative of high precision and recall. In other words, the ability to reject background and distractor faces is correlated, but not directly tied to the closed-universe performance of an algorithm. However, it is useful to compare to existing work, so we include closed-universe accuracy metrics. Using the same experimental setup described in [12], we select 10 random faces from each individual in PubFig83 as testing and use the remaining as training, repeated 5 times. The averages for the features and algorithms listed in Section 3.1.2 along with SVM and LASRC algorithms run on our features (pixels, HOG, Gabor, and LBP) are shown in Table 1.

[12] achieves the highest accuracy with a linear SVM trained using the biologically-inspired HT-L3-1st feature representation. We achieve similar performance (2% less accuracy) with fast, standard, easy-to-extract features compared to HT-L3-1st representations, while using 50X less data (HT-L3-1st uses 200 KB per face at 51,200 dimensions, whereas LASRC uses only 4 KB at 1,024 dimensions). Although LASRC takes an additional 2% drop in accuracy, it performs better than an SVM with baseline features (V1-like-Plus) and the commercial software from face.com. More importantly, we have found better closed-universe performance does not always translate into better rejection of distractors in an open-universe scenario, as we will examine.

4.2. Open-Universe Evaluation Metrics

Since accuracy only compares performance on the test set and does not report how well an algorithm rejects background and distractor faces, another more representative metric for evaluation is required to capture performance. Comparison of open-universe performance, where distractor faces must be rejected, is often done with precision and recall (PR) curves. Recall is defined as the percentage of known faces labeled by a given method with a particular threshold; thus 100% recall means all known faces in the test set have been assigned a label, regardless of whether that label is correct or not. Precision is the number of faces that were correctly identified dived by the number of total faces that were assigned a label (distractor faces not being assigned a label do not count towards precision). Intuitively, one can think of this definition of precision and recall as precision representing the accuracy achieved while labeling some percentage of the known faces (recall).

To summarize precision/recall curves into two numbers that are most useful for real-world operation, we report both the Average Precision (AP) and Recall at 95% Precision. AP represents the overall goodness of the classification and rejection of distractors across a range of operating configurations. Because consumer applications typically value high precision (fewer mistakes) over recall (more tags, possibly incorrectly identified), recall at 95% precision is a particularly good metric. It shows how much of the total test gallery can be labeled while maintaining an accuracy of only 1 mistake out of 20 assigned labels.

To avoid the messiness and clutter of putting all evaluations on a single PR graph, we instead divide them into categories. In each category, we include the top-performing research method, LASRC, for comparison. We will release the PR curves on our website so researchers can run comparisons against any subset of algorithms, or their own.

4.3. Research Algorithms

The precision and recall curve for research algorithms is shown in Fig. 3 with Table 2 showing recall at 95% precision and the average precision for each algorithm. The results of evaluating research algorithms on the PubFig83+LFW dataset are similar to the ones presented in [9], although it is interesting to note that despite having fewer individuals to identify, the PubFig83+LFW dataset is more challenging. This is likely due to the pruning of duplicate and near duplicate images by Pinto *et al*. [12] and having a greater percentage of distractor faces with fewer individuals in the gallery. Algorithms requiring an approximation parameter (KNN-SRC, LLC, and LASRC) used a value of $K = 64$. All algorithms classify in real-time (>30 Hz). Sparse Representation-based Classification (SRC) is very slow for large datasets, so we omitted it as being unsuitable for web-scale tasks, and used fast approximations instead (KNN-SRC and LASRC). We expect SRC to perform slightly better than LASRC.

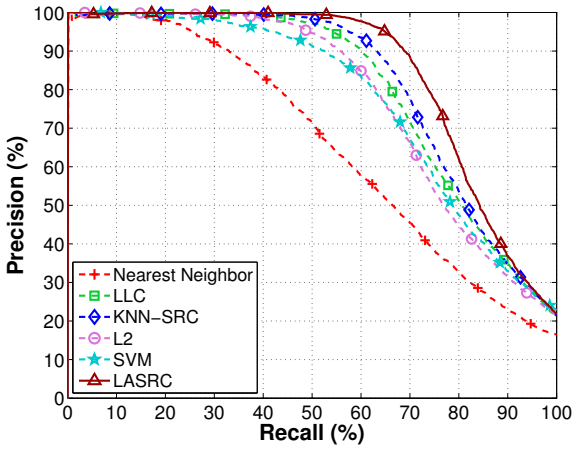


Figure 3. Research algorithm performance in open-universe on PubFig83+LFW: NN, LLC, KNN-SRC, L2, SVM, and LASRC.

Algorithm	Recall (%)	AP (%)
Nearest Neighbor (NN)	25.6	65.9
LLC	54.4	80.2
KNN-SRC	57.5	81.6
L2	49.4	77.9
SVM	41.8	77.6
LASRC	**64.9**	**84.4**

Table 2. Recall at 95% precision and average precision (AP) for research algorithms.

Overall, LASRC outperforms SVMs and other recently introduced algorithms based on linear representations because its approximation using linear regression is more correlated than nearest neighbors (e.g. KNN-SRC and LLC). It is particularly interesting to note that SVMs perform best in closed-universe scenarios (Table 1) while in realistic, open-universe scenarios, LASRC's ability to reject distractors results in higher performance than SVM (Table 2). We hypothesize the sparsity concentration index (SCI) metric of SRC provides a good measure of class confidence.

4.4. Client-Side Libraries

The precision and recall curve for client-side libraries is shown in Fig. 4 with Table 3 listing recall at 95% precision and the average precision for each algorithm. Since OpenCV does not do internal alignment like OpenBR and PittPatt, we include both unaligned and aligned (with PittPatt fiducials) results. It is interesting that simple eye-alignment provides a substantial 12.5% boost in recall. OpenBR handles alignment internally, but still performs poorly, perhaps because it is very new, beta software that is primarily setup to handle face verification tasks (determin-

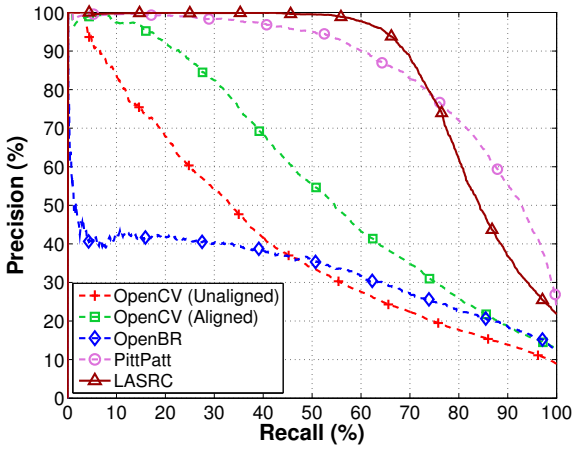


Figure 4. Client-side algorithm performance in open-universe: OpenCV, OpenBR, and PittPatt compared to LASRC. OpenCV is reported both without alignment and with the same alignment as in the research section using PittPatt fiducial locations.

Algorithm	Recall (%)	AP (%)
OpenCV (Unaligned)	4.0	41.6
OpenCV (Aligned)	16.5	57.4
OpenBR	0.1	32.9
PittPatt	50.1	**85.5**
LASRC	**64.9**	84.4

Table 3. Recall at 95% precision and average precision (AP) for client-side libraries and LASRC.

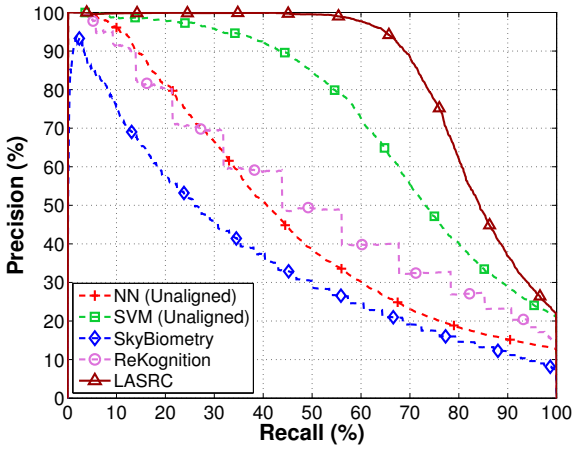


Figure 5. Cloud-based API performance in open-universe of compared to LASRC and simple, unaligned nearest neighbor and SVM algorithms with local features (HOG+LBP+Gabor).

Algorithm	Recall (%)	AP (%)
NN (Unaligned)	11.1	47.5
SVM (Unaligned)	31.6	72.6
SkyBiometry	-	36.3
ReKognition	9.2	52.6
LASRC	**64.9**	**84.4**

Table 4. Recall at 95% precision and average precision (AP) for cloud APIs compared to LASRC and simple, unaligned nearest neighbor and SVM techniques that do not use sophisticated alignment.

ing if two faces are same/not same) rather than identifying faces with a confidence. The proprietary PittPatt face recognition system outperforms all others including LASRC in average precision, but suffers at high precision, with recall trailing LASRC by 14.8% at 95% precision. So depending on if precision or recall is preferred, LASRC or PittPatt serve different needs, with LASRC providing high confidence in identity predictions, but making relatively fewer predictions than PittPatt.

4.5. Cloud-Based APIs

The precision and recall curve for cloud-based APIs is shown in Fig. 5 with Table 4 listing recall at 95% precision and the average precision for each algorithm. It is noteworthy that SkyBiometry and ReKognition have significantly worse performance than many other algorithms. In fact, as a test, we ran nearest neighbor and SVM with our standard Pixels+HOG+LBP+Gabor features *without any alignment* and found similar or better performance than cloud-based APIs. In other words, face identification using standard features and machine learning algorithms without any alignment performs better than SkyBiometry and ReKognition. We hypothesize that the newness of SkyBiometry, the focus on smaller datasets, and the limitation of only allowing only 1,000 faces to be used for training (e.g. only 1/8 of the training faces used by all other algorithms) may all be contributing factors. ReKognition uses all training images and does better, although exhibits a step-wise PR curve because the returned confidences are limited to two significant figures. However, it is still only on par with nearest neighbor without alignment. One hypothesis is that online APIs are designed to be lightweight, preferring speed over accuracy.

4.6. Consumer Applications

Fig. 6 shows distinct points on a PR curve for consumer applications that correspond to individual face recognition settings. For Microsoft Photo Gallery, we used a registry setting to change the recognition between three values: stringent, normal, and loose. Google Picasa was tested with a suggestion threshold of 0.85, 0.80, 0.70, and 0.5. Upon evaluating Google Picasa, we noticed an interesting phenomena: if test faces and distractors were imported into Picasa individually, the results differed markedly compared to if both test faces and distractors were imported at the same time. If each individual face was being classified separately

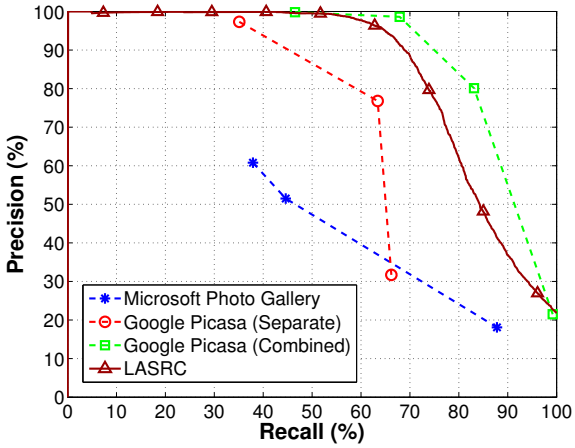


Figure 6. Consumer application performance in open-universe compared to LASRC on PubFig83+LFW, where each data point represents an individual setting and straight lines are drawn between these settings to give an outline of the PR curve.

and individually in Picasa, the results should be very similar. The surprising difference between these two scenarios indicates that Google Picasa is most likely modeling and applying probability distributions of labeled faces to incoming images. Thus, adding the test faces and distractor faces simultaneously allows Picasa to reason more globally about the distribution of identities instead of considering only one individual face at a time. Compared to consumer applications, LASRC outperforms Microsoft Photo Gallery and Google Picasa (Separate), but trails behind Google Picasa (Combined), indicating more global reasoning about identities may be useful in such real-world scenarios.

5. Conclusion

We have investigated the state of face recognition across a broad segment of the field, from research algorithms to consumer products to cloud-based solutions. The findings of our experimentation have lead to the following key insights: (a) Research algorithms are often difficult to get running properly, but offer the greatest flexibility and high performance; (b) There is a lot of variability in both client-side software and consumer applications, although off-the-shelf SDKs and software can do quite well (e.g. PittPatt and Google Picasa); (c) Cloud-based APIs off-load CPU and storage requirements for easy integration, but may be less powerful because of training, storage, and usage limits. Finally, we demonstrated the importance of evaluating in open-universe scenarios with real-world databases to measure face recognition performance on the web. Future work should focus on expanding the scope to include a more comprehensive benchmark for face identification systems.

References

[1] T. Ahonen, A. Hadid, and M. Pietikäinen. Face description with local binary patterns: Application to face recognition. *TPAMI*, 2006. 4

[2] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. Liblinear: A library for large linear classification. *JMLR*, 2008. 4

[3] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, University of Massachusetts, Amherst, 2007. 2

[4] B. Klare, M. Burge, J. Klontz, R. Vorder Bruegge, and A. Jain. Face recognition performance: Role of demographic information. *TIFS*, 2012. 4

[5] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. *ICCV*, 2009. 2

[6] K. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *TPAMI*, 2005. 1

[7] A. R. Martinez and R. Benavente. The ar face database. Technical report, Computer Vision Center (CVC), 1998. 1

[8] Z. Nan and Y. Jian. K nearest neighbor based local sparse representation classifier. In *CCPR*, 2010. 2, 4

[9] E. G. Ortiz and B. C. Becker. Face recognition for web-scale datasets. *CVIU*, in submission. 2, 4, 6

[10] P. J. Phillips, H. Moon, S. Rizvi, and P. J. Rauss. The feret evaluation methodology for face-recognition algorithms. *TPAMI*, 2000. 1

[11] P. J. Phillips, W. T. Scruggs, A. J. OToole, P. J. Flynn, K. W. Bowyer, C. L. Schott, and M. Sharpe. Frvt 2006 and ice 2006 large-scale results. Technical report, National Institute of Standards and Technology, Tech. Rep. NISTIR 7408, 2007. 2

[12] N. Pinto, Z. Stone, T. Zickler, and D. Cox. Scaling up biologically-inspired computer vision: A case study in unconstrained face recognition on facebook. In *CVPR*, 2011. 2, 5, 6

[13] Q. Shi, A. Eriksson, A. van den Hengel, and C. Shen. Is face recognition really a compressive sensing problem? In *CVPR*, 2011. 2, 4

[14] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *CVPR*, 1991. 2

[15] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, 2010. 2, 4

[16] L. Wolf, T. Hassner, and Y. Taigman. Effective unconstrained face recognition by combining multiple descriptors and learned background statistics. *TPAMI*, 2011. 2

[17] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *TPAMI*, 2009. 2, 4

[18] H. Zhang, A. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *CVPR*, 2006. 2