Toward efficient, privacy-aware media classification on public databases[*]

Giulia Fanti
EECS Dept., U.C. Berkeley
Berkeley, California, USA
gfanti@eecs.berkeley.edu

Matthieu Finiasz
CryptoExperts
Paris, France
finiasz@gmail.com

Gerald Friedland
International Computer Science Institute
Berkeley, California, USA
fractor@icsi.berkeley.edu

Kannan Ramchandran
EECS Dept., U.C. Berkeley
Berkeley, California, USA
kannanr@eecs.berkeley.edu

ABSTRACT

The ability to search databases by providing multimedia examples of voices, faces, or locations instead of textual descriptions can be tremendously useful. At the same time, uploading media for queries—especially media that contains sensitive content—means sharing private information with a potentially untrusted service provider. The growing field of privacy-preserving database searches attempts to resolve this tension. Within this scope of private searches, private *media* classification and retrieval is particularly challenging due to the inherent inexactness of recognition; to be useful, image or other media classification systems must identify approximate matches rather than just exact ones. This is difficult to reconcile with distortion-intolerant and resource-heavy privacy primitives, especially in web-scale databases. In this paper, we present an architecture for media classification on public databases that preserves client privacy while achieving asymptotic communication and computation costs that are sublinear in the size of the database. We demonstrate the usefulness of this architecture in the context of a privacy-preserving face recognition system. We observe order-of-magnitude speedups over state-of-the-art private face recognition systems.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query formulation; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*; K.4.1 [**Public Policy Issues**]: Privacy

General Terms

Privacy, media classification

Keywords

Privacy-preserving image classification, private information retrieval, nearest-neighbor search

1. INTRODUCTION

Query-by-example tools allow clients to search a database by uploading examples of intended results. Media domain applications of these tools (e.g., user voice authentication, image similarity searches, or location estimation from camera input) are becoming increasingly widespread. Due to resource constraints on devices like mobile phones, these queries are often outsourced to externally-managed servers for speed and efficiency. However, queries to servers can reveal detailed and potentially sensitive information about clients; search engine data leaks over the past decade have made this abundantly clear [1, 2]. The information richness of queries poses significant privacy concerns, insomuch as client behavior is being monitored and monetized to an unprecedented degree [3, 4, 5]. The problem arises in part because this information can be viewed by several parties, including mobile service providers that communicate the requests, hackers that illegally access server records, government bodies that legally do so, and of course the server itself.

For these reasons, it is important to develop tools that enable servers to process queries-by-example in a privacy-preserving manner. In particular, our problem of interest is as follows (Figure 1): A client possesses a noisy media object (e.g. a facial image captured in uncontrolled conditions), while a clean version of the object is stored in a server-maintained database. The client wishes to learn the identity of the query without revealing any information about the query to the server. This classification task is a special case of "soft" private queries.

[*]This work is supported in part by National Science Foundation grants CNS-1065240 and CCF-0964018, and by NSF Graduate Research Fellowship DGE-1106400. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors or originators and do not necessarily reflect the views of the NSF.

ICMR '14, April 01 - 04 2014, Glasgow, United Kingdom
Copyright 2014 ACM 978-1-4503-2782-4/14/04 $15.00
http://dx.doi.org/10.1145/2578726.2578733.

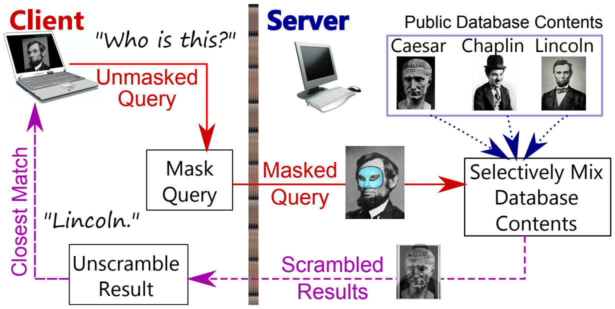

Figure 1: Block diagram of the problem. The client wishes to classify the media query without revealing the query to the server.

Significance: There are some important applications for this kind of technology. On the commercial side, there is demand for privacy-preserving recommendation systems; clients browsing the web could receive relevant advertisements without revealing their preferences to third-party advertisers. On the medical front, medical signals like DNA sequences could be classified against a publicly-maintained database to diagnose irregularities. Another increasingly important area of interest is privacy-conscious surveillance. The FBI recently announced plans to implement a billion-dollar program to locate wanted persons by scanning faces in surveillance video and comparing them to a national database [1]. This program enables an unprecedented level of government surveillance that could easily be used to track citizens' movements. A privacy-preserving image classification algorithm would enable such surveillance to target criminals without invasion of privacy.

In particular, we envision a collaborative approach to surveillance, in which public venues (stores, banks, etc.) with cameras could receive compensation for running face extraction software on surveillance video. The extracted faces would be compared in a private fashion against a government database of criminals; if a customer is not in the database, no matches will return, and the government will learn no information about the query. If the individual *is* a wanted criminal, then the software would notify authorities. This could be cheaper than centrally maintaining surveillance infrastructure and it would prevent police agencies from having unfettered access to the movements of the population.

Contributions: The main technical contribution of this paper is a framework for one-way privacy-preserving media classification on public databases.[1] Our proposed framework approximates Euclidean-distance nearest-neighbor searches, and we apply it to existing Euclidean nearest-neighbor face-recognition algorithms like Eigenfaces and Fisherfaces. The novelty of our work stems partially from using exact information retrieval tools to solve an inexact search problem. However, the broader value of our work is the observation that even using off-the-shelf privacy primitives, one-way-private classification can be *significantly* faster than existing two-way-private counterparts. This suggests that one-way private queries require their own tools and should not be treated as a sub-category of two-way private searches, as is the current norm in the research community.

2. RELATED WORK

There has been a great deal of work on variants of this problem. For instance, when a client knows exactly which file is desired from a database, techniques like private information retrieval or oblivious transfer enable privacy-preserving queries [6, 7]. However, most queries are less precise and consist of searches for files that contain a keyword or resemble a query in some inexact way. Solving such a problem in the private domain is challenging in part because cryptographic primitives are intolerant of distortion (similar numbers become dissimilar in encrypted space), while data recognition and classification requires robustness to noise.

To address this challenge, privacy-preserving data classification has become a popular research topic in recent years. This body of work includes privacy-preserving machine learning tools such as logistic regression [8] and support vector machines (SVM) [9], while [10] gives a nice survey of privacy-preserving nearest-neighbor methods. There is also a great deal of application-centric research focusing on privacy-preserving media classification (a special case of soft signal classification). Work in this area includes authentication and identification of biometrics like faces, fingerprints, or ECG signals [11, 12, 13, 14, 15, 16], as well as video analysis [17], to name a few. Algorithms in this space tend to be heavy, with communication and computation that are asymptotically linear in the database size, and large asymptotic constants. This inefficiency stems partially from computationally-heavy cryptographic primitives that require modular arithmetic and/or communication-heavy primitives like garbled circuits. These tools are necessary because existing research has almost exclusively been restricted to private databases—that is, the client should learn nothing about the database beyond the query's result (in addition to the server learning nothing about the client's query).

We depart from the existing body of work by addressing private search over *public* databases; we believe this problem will become increasingly relevant as media databases like YouTube, Flickr, and Google Images grow in scope. Abandoning the realm of private databases allows us to use privacy primitives like private information retrieval (PIR) [6], which can give significant efficiency gains in practice. To the best of our knowledge, there are only two existing works on private media queries over public databases: a high-level discussion of the problem in [18], and a private image similarity search tool by Shashank *et al.* [19]. In contrast with [18], we actually implement and test a private search tool, providing valuable practical performance data. Our work differs from [19] in that their system accepts a query image and outputs visually similar images; they have no notion of system accuracy and do not address the classification problem. We observe both high classification accuracy and efficiency.

3. ALGORITHM

In this paper, we focus on Euclidean-distance nearest-neighbor searches as a simple yet effective non-parametric classifier. In general, nearest-neighbor classifiers achieve lower accuracy levels than approaches exploiting sophisticated tools like convex optimization [20]. However, some work over the last five years has revived interest in nearest-neighbor methods as a competitive classification tool [21]. While these techniques may not represent the state-of-the-art, an efficient and private nearest-neighbor search is useful

[1] By "one-way", we mean that only the client's privacy is protected, not the server's.

of its own merit.

Consider a classic nearest-neighbor search over a feature space describing media objects. These media objects can include video, image, audio, or even arbitrary data requiring 'soft' matches, like DNA sequences. Traditionally, a server stores one feature vector for each database entry. Upon receiving a (noisy) query feature vector **b** from the client, the server declares the closest database feature under some metric a match; call this closest match vector $\mathbf{a}^*$.

In the private domain, this approach is unacceptable because the server learns the client's query. It is also expensive to compare distances between vectors in a private manner (see e.g. [13, 16]). To make the algorithm privacy-preserving, we alter the search algorithm to search for partial exact matches. Our approach is based on the audio search scheme of Haitsma and Kalker [22]. While their algorithm is not privacy-preserving, it is conducive to integration with standard privacy primitives. We will now describe a non-private variation of their algorithm that will prove useful.

Instead of sending the client's query feature vector **b** directly to the server, we divide the feature vector into chunks of k bits, called subfingerprints. The client starts by sending the server the first subfingerprint from its noisy feature vector. The server returns the feature vectors of all database entries that contain a substring of k bits that match the query's first k bits exactly, i.e. the subfingerprints are identical. Then the *client* computes the Hamming distances of the returned features from the noisy query and chooses the result with the minimum distance (see Figure 2). After doing this for all the subfingerprints in **b** (suppose there are n_s subfingerprints in total, collectively called a fingerprint block), the client learns the nearest neighbor identity $\mathbf{a}^*$. This algorithm succeeds as long as there is at least one subfingerprint that matches exactly between $\mathbf{a}^*$ and **b**; the probability of this occurring depends on the subfingerprint size k. Alternatively, if the client-side noise statistics are known, one can reduce communication and computation by accepting any feature vector within a threshold bit error rate (BER) as a match. This obviously gives lower recognition rates, but can noticeably reduce resource consumption.

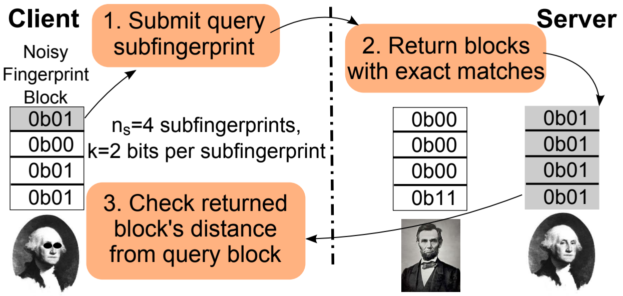
Client
Noisy Fingerprint Block
0b01
0b00
0b01
0b01
1. Submit query subfingerprint
n_s=4 subfingerprints,
k=2 bits per subfingerprint
2. Return blocks with exact matches
Server
0b00
0b00
0b00
0b11
0b01
0b01
0b01
0b01
3. Check returned block's distance from query block

Figure 2: Search toy example over a two-face database. The client submits a subfingerprint query '0b01' and receives all fingerprint blocks containing '0b01' (i.e. only George Washington). The client computes the distance between the returned block and the query (e.g., Hamming distance between the queries), then moves on to the next subfingerprint. The smallest-distance result is declared the closest match.

3.1 Private Search Algorithm

For the private version of this algorithm, instead of sending query subfingerprints in plaintext to the server, we mask both the transmitted subfingerprints and the returned fingerprint blocks using a technique called private information retrieval. This privacy primitive can be integrated easily with the modified algorithm above.

3.1.1 Private Information Retrieval

Private Information Retrieval (PIR) allows a client to retrieve data at a particular index in a database without revealing the query (or the results) to the server. PIR can be done with either a single server or multiple servers storing duplicate copies of the database. Additionally, in the latter case, a minimum number of servers must not communicate to guarantee secrecy; so in a two-server PIR scheme, the servers must not communicate. These anti-collusion requirements are quite strong, so single-server PIR is appealing from a security standpoint. However, multi-server PIR is significantly more efficient in practice than single-server PIR, functioning in some schemes with sublinear asymptotic communication and computation costs [23, 24, 25]. Moreover, single-server PIR (with the exception of trivial database transfer) gives only computational security, while multi-server schemes achieve information theoretic security—this guarantees they are secure against even computationally unbounded adversaries. Also, the anti-collusion requirement could be resolved by storing data on competing cloud services such as Amazon and Google and using secure communication channels between the client and servers to avoid interception. We will address the practicality of this issue again in the conclusion. For these reasons, we believe that multi-server PIR can be more useful than single-server PIR in the long run, and we focus on such schemes in this paper.

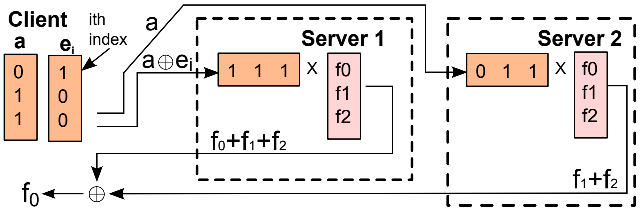

Figure 3: Basic PIR scheme from [6]. Each of two servers computes the bitwise sum of a seemingly-random subset of database files. Because the two user-specified subsets differ only at the ith index, the binary addition of each server's results gives the desired file.

We will now describe the basic, two-server PIR scheme from [6]. Both servers have copies of a database comprised of a sequence of files $f_0, f_1, \ldots, f_n$, and the user wishes to retrieve the ith file, f_i. The user's request can be represented by $e_i \in \{0,1\}^n$, the indicator vector with a 1 at index i and 0's elsewhere. To disguise this query, the user generates a random string $a \in \{0,1\}^n$ with each entry a Bernoulli(1/2) random variable. The queries sent to servers 1 and 2 are $a \oplus e_i$ and a, respectively. Each server computes the inner product of its received query vector with the database using bitwise addition (XOR) and returns the result. The user XORs the results from the two servers to get precisely f_i. The scheme is illustrated in Figure 3.

In an honest-but-curious adversarial model, this multi-server PIR scheme is information theoretically secure. Most private media search schemes rely on single-server, computationally secure primitives. While computational security is currently ubiquitous, it could be neutralized by technological advances like quantum computing, so information-theoretically secure solutions are ultimately safer. As mentioned earlier, these multi-server schemes can also be significantly more efficient than single-server ones; for instance, organizing the database into a d-dimensional cube, where $d \geq 2$ is the number of servers, can reduce the communication to $O(\sqrt[d]{n})$, or $O(\sqrt{n})$ in our two-server example [6]. There exist several other information-theoretically secure PIR algorithms that achieve communication and computation sublinear in database size [23, 25]. In practice, [25] is impractical for standard database sizes due to large constants in the asymptotic costs, but [23] can be practically efficient at the expense of increased data storage. This latter scheme is therefore utilized to understand the asymptotic efficiency of our system.

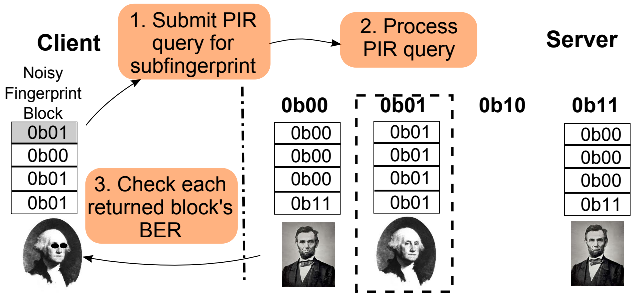

Figure 4: One-way private search scheme example. The two-face database is indexed by subfingerprint. The client submits a PIR query for the first query subfingerprint '0b01' and receives all fingerprint blocks containing '0b01' (i.e. only George Washington). Note that there are actually multiple servers (not depicted), since we are using multi-server PIR. The client computes the BER for the returned block, then moves on to the next subfingerprint.

3.1.2 PIR Integration

To make use of PIR, we first modify the database to have an inverted structure, indexed by subfingerprints; for k-bit subfingerprints, the database has 2^k entries. At the ith index, corresponding to the k-bit representation of subfingerprint i, the database stores all feature vectors containing i as one of their subfingerprints. If the query's first subfingerprint takes on value i, the client submits a PIR query for index i, and receives from the servers all feature vectors that contain subfingerprint i. This scheme is illustrated in Figure 4 for a face recognition example. This search algorithm is information-theoretically secure for the client—a fact that follows from the information-theoretic security of the PIR scheme.

3.1.3 Noise Robustness

As presented, this search scheme implicitly assumes some level of feature quantization that yields exact subfingerprint matches. For arbitrary real-valued features in a Euclidean-distance nearest-neighbor search, no such quantization will exist. Thus we can introduce it with a hashing scheme proposed by Yeo *et al.* [26], which is illustrated in simplified form in Figure 5. To approximate the Euclidean distance between two normalized vectors $\mathbf{v}_1$ and $\mathbf{v}_2$, we individually take the inner product of both vectors with a set of randomly drawn vectors and record the sign (+/-) of each inner product. Let $H(v; h)$ denote the hash bit of vector v projected onto random vector h. If the Euclidean distance between $\mathbf{v}_1$ and $\mathbf{v}_2$ is δ, then for any random vector h, the probability $p_{\mathbf{v}_1,\mathbf{v}_2}$ that $\mathbf{v}_1$ and $\mathbf{v}_2$ have different hash bits is

$$p_{\mathbf{v}_1,\mathbf{v}_2} = P(H(\mathbf{v}_1; h) \neq H(\mathbf{v}_2; h)) = \frac{\pi}{2} \sin^{-1} \frac{\delta}{2}. \qquad (1)$$

Therefore, by projecting all feature vectors onto a fixed set of random vectors, the Hamming distance between the resulting strings of hash bits gives a probabilistic estimate of the Euclidean distance between the vectors. In particular, the expected Hamming distance of the hash vectors is a monotonically increasing function of the Euclidean distance between the vectors. To connect this with the search algorithm, we generate $M = n_s \cdot k$ random projection vectors, and reshape the vector of hash bits into a fingerprint block with n_s subfingerprints of k bits each; this fingerprint block acts as a private-search-compatible feature vector and fits into the private search algorithm described earlier. Recall that k determines the size of the database (there are 2^k elements), so if we left the hash bits in vector form (i.e. $k = 1$), the database would only have two entries: 0 and 1. This would cause the downlink communication from PIR to be prohibitively high. Note that the original feature vectors in Fisherfaces are not normalized, which is a prior assumption for the hashing scheme. Therefore, we normalize the Fisherfaces feature vectors, which leads to small accuracy losses.

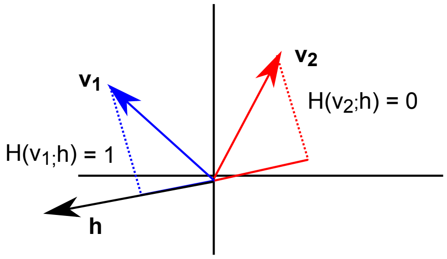

Figure 5: Hashing algorithm for approximating Euclidean distance with a Hamming distance. h is a random vector, while $\mathbf{v}_1$ and $\mathbf{v}_2$ are two arbitrary, normalized vectors. The hash bit is the sign of the inner product of $\mathbf{v}_i$ and h, for $i \in \{1, 2\}$.

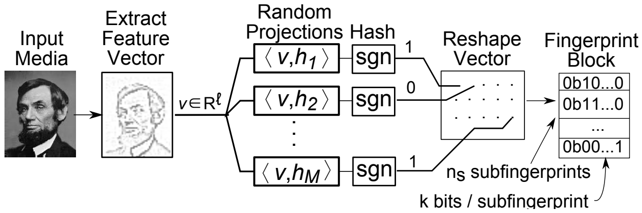

Figure 6: Conversion of arbitrary feature vectors into subfingerprints. h_i denotes a randomly-drawn ℓ-dimensional hyperplane, where ℓ is the original feature vector dimension.

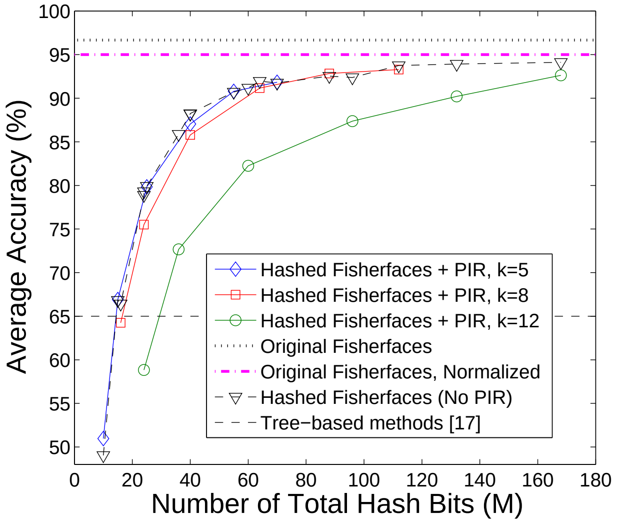

Figure 7: Mean recognition rate as a function of total number of hash bits, parameterized by subfingerprint length k.

4. CASE STUDY: FACE RECOGNITION

To demonstrate our framework, we designed a face recognition system that relies on random hashes combined with various face recognition algorithms. For a standard Euclidean-distance nearest-neighbor search, we used the Eigenfaces and Fisherfaces algorithms [27, 28]. Fisherfaces is a face recognition algorithm that projects vectorized images onto a carefully selected basis, chosen to give robustness to variables like lighting conditions, facial expressions, and occlusions like hair and glasses. This scheme requires the client to learn the basis of Fisherfaces in plaintext. We assume this is acceptable for a public database, but in reality, a server might not want to reveal this information. More generally, the feature extraction process in a media classification scheme might require knowledge of model parameters that the server would rather keep private.

After projecting all the database and query vectors onto this Fisherface basis, the server finds the nearest database neighbor in Euclidean distance to the query feature. Recognition rates using Fisherfaces do not compare to the state-of-the-art, but our objective here is simply to demonstrate the adaptation of a well-known image recognition algorithm to the private domain. For a privacy-preserving version of this algorithm, we extract the Fisherfaces feature vectors and process them using random hashing, as shown in Figure 6. This gives fingerprint blocks of n_s subfingerprints that fit directly into the PIR-based search described earlier.

To evaluate this framework, we provide both analytical and experimental observations. For the latter, we implemented a private face recognition tool in Python, simulating two servers on a single machine. This was run using the ATT database of faces [29]. This database consists of 40 faces over a spread of 10 images each, totaling 400 images. Tests were run on an Intel Core i7-620M processor with two 2.67 GHz processing cores and 4 GB of RAM.

4.1 Accuracy

With respect to classic Fisherfaces, accuracy levels are impacted by three factors: normalization of feature vectors, the hashing scheme, and the division of hash bits into subfingerprints. Recall that our hashing scheme assumes that the input vectors have unit norm; this is not the case in practice, so we normalize our feature vectors to make the mapping between Euclidean distance and Hamming distance hold. This leads to minor losses in algorithm accuracy, which we quantify. The random hashing scheme can be driven to obtain recognition rates that approach the nearest-neighbor accuracy by increasing the total number of random hash vectors $M = n_s \cdot k$. Using the properties of the random hashes, we can also determine the probability of finding an exact subfingerprint match as a function of the number of subfingerprints n_s and subfingerprint length k. If $\mathbf{b}$ is the noisy query vector and $\mathbf{a}^*$ is the correct closest match in the database, then $p_{\mathbf{b},\mathbf{a}^*}$ is the bit error rate between the two hash vectors. The probability of classification success is lower bounded by the probability of seeing at least one exact match (let S denote this event), which is given by

$$P(S) \geq 1 - (1 - (1 - p_{\mathbf{b},\mathbf{a}^*})^k)^{n_s}. \tag{2}$$

From this expression, it follows that shorter subfingerprints (smaller k) and more subfingerprints (larger n_s) increase the probability of finding an exact match. In general, it is difficult to estimate this probability accurately because $p_{\mathbf{b},\mathbf{a}^*}$ cannot be determined a priori and depends on the noise in the system. In an uncontrolled environment, this noise can additionally vary significantly between subjects. However, assuming some upper bound on the noise parameter based on empirical measurement allows us to lower bound the probability of finding an exact match, and therefore the probability of correct classification.

Figure 7 gives empirical accuracy levels using both classic Fisherfaces and hashed Fisherfaces with PIR (our proposed scheme). The dataset was randomly split into 70 percent training data and 30 percent test data. We included accuracy measurements for the hierarchical scheme in [19], though this comparison is unfair since their system is not designed for classification. We observe that their tree-based methods have low classification accuracy, making them ill-suited to our problem.

To evaluate our proposed scheme, Figure 7 shows the slightly reduced accuracy of Fisherhaces when the feature vectors are normalized. It also shows accuracy levels for hashed Fisherfaces without PIR (i.e. using the hash bits as features without partitioning them into subfingerprints); this indicates how much accuracy is lost due purely to feature hashing as opposed to the division of hash bits into subfingerprints. For small M, hashing appears to significantly reduce the overall accuracy, but increasing the number of total hash bits drives recognition rates toward the non-private rates. For a fixed M, larger subfingerprints (i.e. larger k) lower recognition rates by reducing the probability an exact match. However, we observe that for a fixed number of total hash bits, subfingerprints as large as 5 or 8 bits give nearly optimal accuracy levels compared to hashed Fisherfaces without PIR.

This is significant because given feature vectors with inherent quantization and a good recognition rate, a PIR-based search scheme need not reduce search accuracy significantly. Of course, the degree of accuracy degradation depends on system noise, but feature quantization at least gives us a mechanism for tuning sensitivity to noise. Some

Algorithm	**Privacy Scheme**	**Communication**	**Computation**
Hashed Fisherfaces	2-server PIR [23]	$O(\sqrt[3]{n})$	$O\left(n/\log^2 n\right)$
Hierarchical [19]	2-server PIR [23]	$O(\sqrt[3]{n})$	$O\left(n/\log^2 n\right)$
Eigenfaces [16]	HE, Garbled Circ. [30, 31]	$O(n)$	$O(n)$
Eigenfaces + Backtracking [14]	HE, Garbled Circ. [30, 31]	$O(n)$	$O(n)$

Table 1: Face recognition asymptotic communication and computation costs for privacy-preserving face recognition; 'Hashed Eigenfaces' refers to our suggested scheme, and we also compare this to the online complexities of two benchmark face recognition systems [16, 15]. We use $k = \log n$, where k is subfingerprint length and n is database size.

examples of features with these properties include the facial profile features by Osadchy *et al.* [15] and the audio features in Haitsma and Kalker's audio recognition system [22].

4.2 Communication and Computation

Parameter selection can impact algorithm efficiency significantly, both in terms of communication and computation costs. We will discuss asymptotic and experimental results.

4.2.1 Asymptotic Results

The asymptotic communication cost of this search algorithm is $O(\max(p(2^k; B), m(n))$, where n is the database size and $m(n)$ is the expected number of exact subfingerprint matches in the database; k is the number of bits in each subfingerprint, and $p(2^k)$ is the total communication complexity of a PIR search on a list of k-bit subfingerprints (i.e. database with 2^k entries).

There is a tradeoff between $m(n)$ and $p(2^k)$ since subfingerprint size determines the expected number of matches for a fixed database size. For an asymptotic cost comparison, we choose k as $O(\log n)$, so the expected number of matches scales as $O(1)$ with database size; the dominant communication cost consequently comes from uplink PIR queries. This choice is not necessarily optimal for communication, but it does reduce the client-side computation to $O(1)$, which is important since the client is assumed to have limited resources. We use this assumption on k in successive cost calculations, but we will also address the question of how to choose k given a total number of hash bits. Asymptotic computation costs are also dominated by PIR, and therefore depend on the PIR scheme.

Comparison with other algorithms is challenging, since our work is the first to directly address private media classification on public databases (as far as we know). For completeness, Table 1 gives asymptotic communication and computation costs for our face recognition scheme and three benchmark schemes private media search schemes [19, 16, 14]. The comparison with the last two schemes is unfair since both [16] and [14] guarantee two-way privacy, thereby solving a fundamentally harder problem. Nonetheless, we included these schemes as representative, efficient two-way schemes; the point is that order-level gains in asymptotic efficiency can be had by exploiting one-way privacy. We should point out that linear cost scaling is not inherently problematic; the broader issue is that two-way privacy schemes using cryptographic primitives like homomorphic encryption and garbled circuits incur large constants that limit the practical value of the scheme. The PIR tools used in our framework have a comparatively small constant factor multiplying the asymptotic costs because they rely on bitwise addition.

The only other scheme addressing private media searches over public databases is that of Shashank *et al.* [19]; however, their scheme does not directly address the classification problem, and as such, it has inadequate classification accuracy for our purposes. The efficiency costs for their algorithm were computed using the same multi-server PIR scheme for maximum efficiency [23]. Their asymptotic communication and computation costs are equivalent to ours, and additionally have smaller constants, which is to be expected from a hierarchical database structure. However, note that the scheme in [19] requires $\log n$ rounds of communication, while ours requires at most n_s rounds.

Finally, we note that expanding the database in our model is relatively straightforward. Since the Fisherfaces training phase extracts only the top principal components from the Fisher basis, adding files does not alter the optimal Fisherfaces basis significantly. This means that recognition rates will still be acceptable using the old basis. So to add a database entry, we simply project the new image onto the old basis of Fisherfaces and store the hash bits. The main issue is that after adding enough faces to alter the aggregate database characteristics, the Fisherface basis will need to be re-trained. If the original database is large, this will only happen after adding many new files that diverge statistically from the old database features.

4.2.2 Empirical Results

Many of the applications we have cited will ultimately require the use of large databases, and privacy-preserving techniques are much slower and more resource-hungry than non-private ones. Since our scheme and comparable schemes in the literature are not currently efficient enough for web-scale applications, there is value in considering how to ultimately make these technologies more practical. In particular, we want to understand performance limitations of the system; we explored these scalability issues on synthetic data.

Table 2 gives a breakdown of the computational costs for each phase of the algorithm at different database sizes using a linear PIR scheme; this table takes $k = 5$ and $n_s = 11$. As expected, the online runtime is much smaller than the offline runtime and the online runtime is bottlenecked computationally by the PIR queries; hence the linear cost scaling (we would expect sublinear growth using a sublinear PIR scheme, as in Table 1). To better understand the effect of database size on query response time, Figure 8 provides the average online runtime for an image query using our scheme and the scheme of Huang et al. [14]. This plot was also generated with a linear PIR scheme [6] to emphasize the detrimental effect of modular exponentiation on overall efficiency; even when both schemes have computational costs that scale linearly in the database size, our scheme is an order of magnitude faster. Again, our query runtime would decrease using a sublinear PIR scheme, like [23].

Additionally, it is worth noting that the previous esti-

		Database Size (n)					
		128		512		1024	
	Runtime / Bandwidth	s	KB	s	KB	s	KB
Offline	Feature Extraction	2.9	220 ↓	26.0	280 ↓	76.1	280 ↓
Online	Query Feature Extraction	0.012	0	0.015	0	0.018	0
	PIR Queries	0.20	0.088 ↑ + 9.5 ↓	0.96	0.088 ↑ + 36↓	2.3	0.088 ↑ + 72 ↓
	Online Total	0.21	9.6	0.97	36.1	2.32	72.1

Table 2: Experimental algorithm runtime for the different phases of the algorithm, with $k = 5$, $n_s = 11$. ↑ denotes uplink communication while ↓ denotes downlink communication.

mates rely entirely on serial query processing. However, each PIR query could be submitted in parallel; since PIR is computationally-bottlenecked for large database sizes, this would speed up the response time significantly. Note that tree-based algorithms like that of Shashank *et al.* are not easily parallelizable since each round of communication depends on the previous round [19].

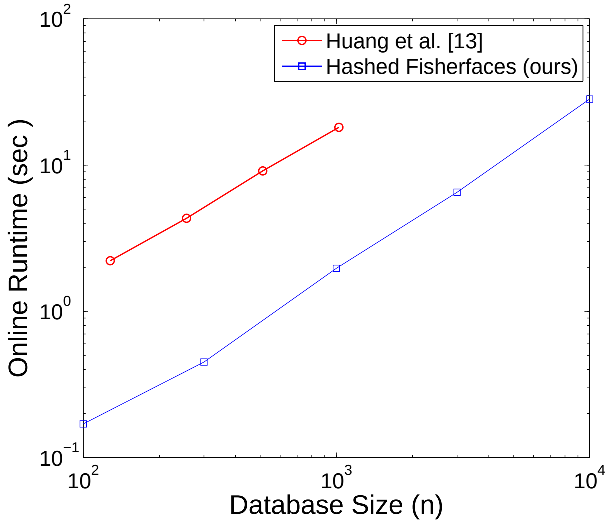

Figure 8: Estimated runtime scaling for practical database sizes. Our projections are compared to estimates for a state-of-the-art two-way private scheme.

5. DISCUSSION

Broadly, our goal is to facilitate the integration of privacy with common technology tools. In this paper, we have presented what we believe to be the first architecture in the literature for private media classification over public databases, and tested the architecture on a face recognition system. As with most privacy-preserving algorithms, our proposed architecture is likely not efficient enough for web-scale databases. However, as the engineering community pushes to develop scalable privacy-preserving tools, it is important to examine other tools besides the usual workhorses of homomorphic encryption and garbled circuits. Indeed, as observed in this work, we gained order-of-magnitude speedups by fitting PIR into the media classification problem.

Important directions: In order to push these technologies to a usable level, there are some important aspects that require more attention. For instance, media fusion (i.e., combining features from a variety of media forms) can boost classification accuracy, but it is particularly challenging in the private domain. Since privacy primitives are heavy, it is important to manipulate as little data as possible, while video processing typically requires higher-dimensional multimedia representations of information [17]. Before implementing our face recognition system, we applied our architecture to music-recognition, implying that audio and visual features are individually tractable in our framework. Combining these features in a compact, quantized manner remains an open problem. The task is nonetheless very important for boosting recognition rates; this would likely be necessary for the video applications discussed in the introduction, for instance.

In a similar vein, we would like to better understand how to modify this architecture to other algorithm classes. In particular, it is not clear that the scheme is limited in applicability to nearest-neighbor searches. With the goal of boosting accuracy, it could be productive to apply similar concepts to potentially more complex search algorithms. In particular, there has been interest recently in privacy-preserving machine learning techniques, such as logistic regression [8].

Another direction for future work focuses on the practicality and efficiency of PIR itself. For instance, one of the most problematic assumptions of multi-server PIR is the anti-collusion requirement. To relax this assumption, there exist PIR schemes that increase the number of servers that need to collude in order to break information-theoretic security [32]. Along these lines, it is important to push for more efficient and robust PIR schemes.

Private media recognition over public databases has the potential to become an important tool, particularly given the current social and technological landscape. It seems that information-theoretic PIR tools could eventually lead to tools efficient enough for integration into the public sphere, particularly in domains like privacy-preserving surveillance or recommendation systems.

6. REFERENCES

[1] M. Barbaro and T. Zeller, “A face is exposed for AOL searcher no. 4417749,” *New York Times Technology*, August 2006, Retrieved from `http://www.nytimes.com/2006/08/09/technology/09aol.html?pagewanted=all`.

[2] D. McCullagh, “AOL’s disturbing glimpse into users’ lives,” *CNET*, August 7 2006.

[3] Google, “Transparency report,” 2013, Retrieved from `http://www.google.com/transparencyreport/userdatarequests/`.

[4] D. Gross, “Yahoo hacked, 450,000 passwords posted online,” *CNN Tech*, Retrieved from `http://www.cnn.`

com/2012/07/12/tech/web/yahoo-users-hacked.

[5] D. McCullagh, “Verizon draws fire for monitoring app usage, browsing habits,” *CNET*, October 16 2012, Retrieved from http://news.cnet.com/8301-13578_3-57533001-38/verizon-draws-fire-for-monitoring-app-usage-browsing-habits/.

[6] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, “Private information retrieval,” in *Proc. IEEE Symposium on Foundations of Computer Science*, Milwaukee, WI, 1995, pp. 41–50.

[7] M.O. Rabin, “How to exchange secrets with oblivious transfer,” *Technical report TR-81, Aiken Computation Lab, Harvard University*, 1981.

[8] K. Chauduri and C. Monteleoni, “Privacy-preserving logistic regression,” in *Proc. 22nd Annual Conf. on Neural Information Processing Systems*, 2008, pp. 289–296.

[9] H. Yu, X. Jiang, and J. Vaidya, “Privacy-preserving svm using nonlinear kernels on horizontally partitioned data,” in *Proceedings of the 2006 ACM symposium on Applied computing*. ACM, 2006, pp. 603–610.

[10] S. Rane and P. Boufounos, “Privacy-preserving nearest neighbor methods: Comparing signals without revealing them,” *Signal Processing Magazine, IEEE*, vol. 30, no. 2, pp. 18–28, 2013.

[11] M. Barni, P. Failla, R. Lazzeretti, A.R. Sadeghi, and T. Schneider, “Privacy-preserving ECG classification with branching programs and neural networks,” *IEEE Trans. on Information Forensics and Security*, vol. 6, no. 2, pp. 452–468, 2011.

[12] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R.D. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, A. Piva, and F. Scotti, “A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates,” in *Proc. IEEE Intl. Conf. on Biometrics: Theory Applications and Systems*, Washington, D.C., 2010, pp. 1–7.

[13] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, “Privacy-preserving face recognition,” in *Privacy Enhancing Technologies*, I. Goldberg and M. Atallah, Eds., vol. 5672 of *Lecture Notes in Computer Science*, pp. 235–253. Springer-Verlag, Berlin, Heidelberg, 2009.

[14] Y. Huang, L. Malka, D. Evans, and J. Katz, “Efficient privacy-preserving biometric identification,” in *Network and Distributed System Security Symposium*, San Diego, CA, 2011.

[15] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, “SCiFI - a system for secure face identification,” in *IEEE Symposium on Security and Privacy*, Oakland, CA, 2010, pp. 239–254.

[16] A.R. Sadeghi, T. Schneider, and I. Wehrenberg, “Efficient privacy-preserving face recognition,” in *Information, Security and Cryptology âĂŞ ICISC 2009*, D. Lee and S. Hong, Eds., vol. 5984 of *Lecture Notes in Computer Science*, pp. 229–244. Springer-Verlag, Berlin, Heidelberg, 2010.

[17] W. Lu, A. Varna, and M. Wu, “Secure video processing: Problems and challenges,” in *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, 2011, pp. 5856–6149.

[18] G. Fanti, M. Finiasz, and K. Ramchandran, “One-way private media search on public databases: The role of signal processing,” *Signal Processing Magazine, IEEE*, vol. 30, no. 2, pp. 53–61, 2013.

[19] J. Shashank, P. Kowshik, K. Srinathan, and C.V. Jawahar, “Private content based image retrieval,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Anchorage, AK, 2008, pp. 1–8.

[20] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 210–227, 2009.

[21] O. Boiman, E. Shechtman, and M. Irani, “In defense of nearest-neighbor based image classification,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[22] J. Haitsma and T. Kalker, “A highly robust audio fingerprinting system,” in *Proc. the Intl. Symposium on Music Information Retrieval*, Paris, France, 2002, pp. 107–115.

[23] A. Beimel, Y. Ishai, and T. Malkin, “Reducing the servers’ computation in private information retrieval: PIR with preprocessing,” *J. Cryptology*, vol. 17, no. 2, pp. 125–151, 2004.

[24] F. Olumofin and I. Goldberg, “Revisiting the computational practicality of private information retrieval,” in *Financial Cryptography and Data Security*, G. Danezis, Ed., vol. 7035 of *Lecture Notes in Computer Science*, pp. 158–172. 2012.

[25] D. Woodruff and S. Yekhanin, “A geometric approach to information theoretic private information retrieval,” in *Proc. IEEE Conf. on Computational Complexity*, San Jose, CA, 2005, pp. 275–284.

[26] C. Yeo, P. Ahammad, H. Zhang, and K. Ramchandran, “Rate-efficient visual correspondences using random projections,” in *Proc. IEEE Intl. Conf. on Image Processing*, San Diego, CA, 2008, pp. 217–220.

[27] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman, “Eigenfaces vs. fisherfaces: recognition using class specific linear projection,” *Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 711–720, 1997.

[28] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71âĂŞ86, 1991.

[29] AT&T Laboratories Cambridge, “The database of faces,” Retrieved from http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html, 2002.

[30] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in *Advances in Cryptology—EUROCRYPT ‘99*, S. Jacques, Ed., vol. 1592 of *Lecture Notes in Computer Science*, pp. 223–238. Springer-Verlag, Berlin, Heidelberg, 1999.

[31] B. Pinkas, “Cryptographic techniques for privacy-preserving data mining,” *ACM SIGKDD Explorations Newsletter*, vol. 4, no. 2, pp. 12–19, 2002.

[32] I. Goldberg, “Improving the robustness of private information retrieval,” in *Security and Privacy, 2007. SP’07. IEEE Symposium on*. IEEE, 2007, pp. 131–148.