MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Secure Sound Classification: Gaussian Mixture Models

Madhusudana V. S. Shashanka, Paris Smaragdis

TR2006-065 May 2006

Abstract

We propose secure protocols for gaussian mixture-based sound recognition. The protocols we describe allow varying levels of security between two collaborating parties. The case we examine consists of one party (Alice) providing data and other party (Bob) providing a recognition algorithm. We show that it is possible to have Bob apply his algorithm on Alice's data in such a way that the data and the recognition results will not be revealed to Bob thereby guaranteeing Alice's data privacy. Likewise we show that it is possible to organize the collaboration so that a reverse engineering of Bob's recognition algorighm cannot be performed by Alice. We show how gaussian mixtures can be implemented in a secure manner using secure computation primitives implementing simple numerical operations and we demonstrate the process by showing how it can yield identical results to a non-secure computation while maintaining privacy.

IEEE International Conference on Acoutiscs, Speech and Signal Processing (ICASSP)

SECURE SOUND CLASSIFICATION: GAUSSIAN MIXTURE MODELS

Madhusudana V. S. Shashanka[*]

Boston University Hearing Research Center
677 Beacon St, Boston MA 02215

Paris Smaragdis

Mitsubishi Electric Research Labs (MERL)
201 Broadway, Cambridge MA 02139

ABSTRACT

We propose secure protocols for gaussian mixture-based sound recognition. The protocols we describe allow varying levels of security between two collaborating parties. The case we examine consists of one party (Alice) providing data and other party (Bob) providing a recognition algorithm. We show that it is possible to have Bob apply his algorithm on Alice's data in such a way that the data and the recognition results will not be revealed to Bob thereby guaranteeing Alice's data privacy. Likewise we show that it is possible to organize the collaboration so that a reverse engineering of Bob's recognition algorithm cannot be performed by Alice. We show how gaussian mixtures can be implemented in a secure manner using secure computation primitives implementing simple numerical operations and we demonstrate the process by showing how it can yield identical results to a non-secure computation while maintaining privacy.

1. INTRODUCTION

In today's highly networked world the problem of data privacy is becoming increasingly relevant. As many researchers working on classification repeatedly observe, accepting data is always welcome but obtaining it is not easy. Legal and security constraints are often hindering open cooperation and make data exchange a cumbersome process (if at all possible). This is especially the case in audio and speech processing where extensive recording databases by large corporations and governments are kept in the dark in fear of privacy or security violations. The same privacy issues also extend in the realm of commercial ventures where the business model of a data processing company analyzing customer data as a service is always greeted with suspicion. In this paper we address this model of processing where privacy of both data and algorithms is a priority of two cooperating, but not trusting, parties. With no loss of generality we specifically concentrate on a gaussian mixture-based sound recognition task. We show that it is indeed possible to have a secure cooperation where there are no privacy issues while the required computations and results take place. The remainder of this paper is ordered as follows. In section 2 we formally introduce the problem at hand, in section 3 we introduce the secure computation primitives that are employed for this task, in section 4 we explain how the secure primitives can be combined to perform various forms of secure classification, and finally in section 5 we present some results before we conclude.

2. PROBLEM FORMULATION

Secure classification allows two parties, Alice and Bob, to engage in a protocol that will allow Alice to classify her data using Bob's classifier without revealing anything to Bob. Also, Alice will learn nothing about the classifier, other than an answer to her query. A two-party protocol between Alice and Bob is *secure* when privacy and correctness are guaranteed for both Alice and Bob. It is said that a protocol *protects privacy* when the information that is leaked by the distributed computation is limited to the information that can be learned from the designated output of the computation. In the *semi-honest* case, both parties follow the protocol as prescribed but may record all messages and subsequently deduce information not derivable solely from the protocol output. In the *malicious* case, however, no assumption is made about the behavior of either party. It is required that the privacy of one party is preserved even in the case of an arbitrary behavior of the second party. A protocol in the semi-honest case can be made secure in the malicious case when accompanied with zero-knowledge proofs that both parties follow the protocol.

3. PRIMITIVES

We use certain primitives in the protocols we present and based on how the primitives are implemented, one can achieve different levels of security and computational/communication efficiency. In general, there is a trade-off between security and efficiency. Below, we describe the primitives that we use and briefly discuss about protocols that implement them.

3.1. Secure Inner Products (SIP)

The primitive which we use most often is for computing secure inner products. If Alice has vector $\mathbf{x}$ and Bob has vector $\mathbf{y}$, a secure inner product protocol produces two numbers a and b such that $a + b = \mathbf{x}^t\mathbf{y}$. Alice will get the result a and

[*] This work performed while at MERL.

Bob will get the result b. To simplify notation, we shall denote a secure inner product computation $\mathbf{x}^t\mathbf{y}$ as $SIP(\mathbf{x}, \mathbf{y})$.

Many protocols have been proposed and they can be categorized as cryptographic protocols (eg. [1, 2]) and algebraic protocols (eg. [3, 4]). They provide different levels of security and efficiency. In this paper, we use a cryptographic protocol based on the idea of homomorphic encryption. See the appendix for a description of the protocol and [2] for a proof that the protocol is correct and secure.

3.2. Secure Maximum Index Protocol ($SMAX$)

Let Alice have a vector $\mathbf{x} = [x_1 \ldots x_d]$ and Bob have the vector $\mathbf{y} = [y_1 \ldots y_d]$, they would like to compute the index of the maximum of $\mathbf{x} + \mathbf{y} = [(x_1 + y_1) \ldots (x_d + y_d)]$. At the end of the protocol, neither party will know the actual value of the maximum. Notice that the same protocol can be used to compute the index of the minimum. We denote this as $j = SMAX(\mathbf{x}, \mathbf{y})$.

Many generic secure two-party protocols have been proposed that enable computation of a function f on the input shares (e.g. [5]). The function f in our case gives the index of the maximum of the sum of the input shares. The communication complexity of most such protocols is linear in the size of the circuit being evaluated.

Another approach is to follow the idea presented in [6]. Bob generates a random polynomial in two variables $f(x, y) = f'(x+y)$ such that $f'(z_i) \leq f'(z_j)$ if and only if $z_i \leq z_j$. For each $i = 1, 2, \ldots, d$, Alice uses OPE (oblivious polynomial evaluation) once to learn $h_i(x_i)$ where $h_i(x) = f(x, y_i)$. The index for which $h_i(x_i)$ is the maximum is the answer Alice is looking for. Notice that neither party will be able to learn the actual value of the maximum element. However, Alice will be able to learn the order of elements in $\mathbf{x} + \mathbf{y}$.

3.3. Secure Maximum Value Protocol ($SVAL$)

Let Alice have a vector $\mathbf{x} = [x_1 \ldots x_d]$ and Bob have the vector $\mathbf{y} = [y_1 \ldots y_d]$, they would like to compute the value of the maximum element in $\mathbf{z} = \mathbf{x} + \mathbf{y}$. After the protocol, neither party will know the index of the maximum element. Notice that the same protocol can be used to compute the value of the minimum. Let us denote this as $a + b = SVAL(\mathbf{x}, \mathbf{y})$.

For this protocol, we can use the idea presented in [7]. Let us first consider a naive approach. Notice that $z_i \geq z_j \iff (x_i - x_j) \geq (y_j - y_i)$. Alice and Bob can do such pairwise comparisons and mimic any standard maximum finding algorithm to learn the value of the maximum. To perform the comparisons securely, they can use a protocol for Yao's millionaire problem [5].

However, if Alice and Bob follow the above naive approach, both will be able to also find the index of the maximum. Hence, the idea is for Alice and Bob to obtain two vectors whose sum is a random permutation of $\mathbf{z}$. Neither Alice nor Bob should know the permutation. They can then follow the above naive approach on their newly obtained vectors to compute additive shares of the maximum element. See [7] for the detailed protocol.

4. SECURE CLASSIFICATION

Alice has a d-component data vector $\mathbf{x}$ and Bob knows multivariate gaussian distributions of N classes $\omega_i, i = \{1, \ldots, N\}$ that the vector could belong to. They would like to engage in a protocol that lets Bob classify Alice's data but neither of them wants to disclose data to the other person. We propose protocols which enable such computations.

The idea is to evaluate the value of the *discriminant function*

$$g_i(\mathbf{x}) = \ln p(\mathbf{x}|\omega_i) + \ln P(\omega_i) \qquad (1)$$

for all classes ω_i and assign $\mathbf{x}$ to class ω_i if $g_i(\mathbf{x}) > g_j(\mathbf{x})$ for all $j \neq i$. Here, $p(\mathbf{x}|\omega_i)$ is the class-conditional probability density function and $P(\omega_i)$ is the *a priori* probability of class ω_i. We consider two cases where: (1) each class is modeled as a single multivariate gaussian, and (2) each class modeled as a mixture of gaussians.

4.1. Case 1: Single Multivariate Gaussian

We assume that the distribution of data is multivariate gaussian i.e. $p(\mathbf{x}|\omega_i) \sim \mathcal{N}(\mu_i, \mathbf{\Sigma}_i)$, where μ_i is the mean vector and $\mathbf{\Sigma}_i$ is the covariance matrix of class ω_i. Ignoring the constant term $(d/2)\ln 2\pi$, we can write equation (1) as:

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\mu_i)^t\mathbf{\Sigma}_i^{-1}(\mathbf{x}-\mu_i) - \frac{1}{2}\ln|\mathbf{\Sigma}_i| + \ln P(\omega_i) \qquad (2)$$

Simplifying, we have:

$$g_i(\mathbf{x}) = \mathbf{x}^t\bar{\mathbf{W}}_i\mathbf{x} + \bar{\mathbf{w}}_i^t\mathbf{x} + w_{i0} \qquad (3)$$

where

$$\bar{\mathbf{W}}_i = -\frac{1}{2}\mathbf{\Sigma}_i^{-1}, \quad \bar{\mathbf{w}}_i = \mathbf{\Sigma}_i^{-1}\mu_i, \quad \text{and}$$
$$w_{i0} = -\frac{1}{2}\mu_i^t\mathbf{\Sigma}_i^{-1}\mu_i - \frac{1}{2}\ln|\mathbf{\Sigma}_i| + \ln P(\omega_i)$$

Let us create the $(d+1)$-dimensional vectors $\bar{\mathbf{x}}$ and $\mathbf{w}_i$ by appending the value 1 to $\mathbf{x}$ and appending w_{i0} to $\bar{\mathbf{w}}_i$. By changing $\bar{\mathbf{W}}_i$ into a $(d+1)\times(d+1)$ matrix $\mathbf{W}_i$ where the first d components of the last row are zeros and the last column is equal to $\mathbf{w}_i^t$, we can express equation (3) in a simplified form:

$$g_i(\mathbf{x}) = \bar{\mathbf{x}}^t\mathbf{W}_i\bar{\mathbf{x}}$$

Expressing $\bar{\mathbf{x}}$ as $\mathbf{x}$ for simplicity, we can write the above equation as:

$$g_i(\mathbf{x}) = \mathbf{x}^t\mathbf{W}_i\mathbf{x} \qquad (4)$$

Henceforth, we shall use $\mathbf{x}$ to denote a $(d+1)$-dimensional vector with the last component equal to 1 unless otherwise mentioned.

4.1.1. *Protocol: Single Multivariate Gaussian (SMG)*

Input: Alice has vector $\mathbf{x}$, Bob has $\mathbf{W}_i$ for $i = 1, 2, \ldots, N$. We express the matrix $\mathbf{W}_i$ as $[\mathbf{W}_i^1 \mathbf{W}_i^2 \ldots \mathbf{W}_i^{d+1}]$, where $\mathbf{W}_i^j$ is the j-th column of $\mathbf{W}_i$.
Output: Alice learns I such that $g_I(\mathbf{x}) > g_j(\mathbf{x})$ for all $j \neq I$. Bob learns nothing about $\mathbf{x}$.

1. For $i = 1, 2, \ldots, N$
   (a) For $j = 1, \ldots, d+1$, Alice and Bob perform $SIP(\mathbf{x}, \mathbf{W}_i^j)$ to obtain the vectors $\mathbf{a}_i = [a_i^1 \ldots a_i^{d+1}]$ and $\mathbf{b}_i = [b_i^1 \ldots b_i^{d+1}]$ respectively. Alice then computes $\mathbf{a}_i\mathbf{x}$.
   (b) Alice and Bob perform $SIP(\mathbf{b}_i, \mathbf{x})$ to obtain q_i and r_i respectively.
2. Alice has vector $\mathbf{A} = [(\mathbf{a}_1\mathbf{x} + q_1) \ldots (\mathbf{a}_N\mathbf{x} + q_N)]$ and Bob has vector $\mathbf{B} = [r_1 \ldots r_N]$.
3. Alice and Bob perform the secure maximum index protocol between the vectors $\mathbf{A}$ and $\mathbf{B}$ and Alice obtains $I = SMAX(\mathbf{A}, \mathbf{B})$.

Correctness: In step 1, $\mathbf{a}_i$ and $\mathbf{b}_i$ are vectors such that $\mathbf{a}_i + \mathbf{b}_i = \mathbf{x}^t\mathbf{W}_i$. Also, $\mathbf{b}_i\mathbf{x} = q_i + r_i$. Hence, $\mathbf{x}^t\mathbf{W}_i\mathbf{x}$ is given by $\mathbf{a}_i\mathbf{x} + q_i + r_i$. I is the value of i for which $\mathbf{x}^t\mathbf{W}_i\mathbf{x}$ is maximum.
Efficiency: For a given $i = I$, the above protocol has $(d+2)$ SIP calls. Hence, it would take $N(d+2)$ SIP calls and one call of $SMAX$.
Security: If Bob gets to know the dot products of d different vectors with $\mathbf{x}$, he can learn $\mathbf{x}$ completely. However, we see that neither Bob nor Alice ever learn the complete result of any dot product. Hence, if the protocols for SIP and $SMAX$ are secure, the above protocol is secure.

4.2. Case 2: Mixture of Gaussians

Let us now consider the case where each class is modeled as a mixture of gaussians. Let the mean vector and covariance matrix of the j-th gaussian in class ω_i be μ_{ij} and $\mathbf{\Sigma}_{ij}$ respectively. Hence we have $p(\mathbf{x}|\omega_i) = \sum_{j=1}^{J_i} \alpha_{ij}\mathcal{N}(\mu_{ij}, \mathbf{\Sigma}_{ij})$ where J_i is the number of gaussians describing class ω_i and α_{ij} are the mixture coefficients. The log likelihood for the j-th gaussian in the i-th class is given by

$$l_{ij}(\mathbf{x}) = \mathbf{x}^t\bar{\mathbf{W}}_{ij}\mathbf{x} + \bar{\mathbf{w}}_{ij}^t\mathbf{x} + w_{ij} \tag{5}$$

where

$$\bar{\mathbf{W}}_{ij} = -\frac{1}{2}\mathbf{\Sigma}_{ij}^{-1}, \qquad \bar{\mathbf{w}}_{ij} = \mathbf{\Sigma}_{ij}^{-1}\mu_{ij}, \qquad \text{and}$$
$$w_{ij} = -\frac{1}{2}\mu_{ij}^t\mathbf{\Sigma}_{ij}^{-1}\mu_{ij} - \frac{1}{2}\ln|\mathbf{\Sigma}_{ij}|$$

Expressing $\mathbf{x}$ as a $(d+1)$-dimensional vector and $\bar{\mathbf{W}}_{ij}$, $\bar{\mathbf{w}}_{ij}$, w_{ij} together as the $(d+1) \times (d+1)$ matrix $\mathbf{W}_{ij}$ as done in the previous case, we can simplify equation (5) as:

$$l_{ij}(\mathbf{x}) = \mathbf{x}^t\mathbf{W}_{ij}\mathbf{x} \tag{6}$$

Hence, the discriminant function for the i-th class can be written as

$$\begin{aligned} g_i(\mathbf{x}) &= \text{logsum}\big(\ln\alpha_{i1} + l_{i1}(\mathbf{x}), \ldots, \ln\alpha_{iJ_i} + l_{iJ_i}(\mathbf{x})\big) \\ &\quad + \ln P(\omega_i) \qquad \text{where} \end{aligned} \tag{7}$$

$$\text{logsum}(x_1, \ldots, x_{J_i}) = \max(x_1, \ldots, x_{J_i}) + \ln\big(\sum_{j=1}^{J_i} e^{\Delta_j}\big),$$
$$\Delta_j = x_j - \max(x_1, \ldots, x_{J_i}) \qquad \forall j \in \{1, \ldots, J_i\}.$$

4.2.1. *Protocol: Mixture of Gaussians*

Input: Alice has vector $\mathbf{x}$, Bob has $\mathbf{W}_{ij}$, α_{ij} and $P(\omega_i)$ for $i = 1, 2, \ldots, N$, and $j = 1, 2, \ldots, J_i$.
Output: Alice learns I such that $g_I(\mathbf{x}) > g_j(\mathbf{x})$ for all $j \neq I$. Bob learns nothing about $\mathbf{x}$.

1. For $i = 1, 2, \ldots, N$
   (a) Alice and Bob engage in steps 1 and 2 of Protocol 1 for the J_i gaussians in the i-th mixture to obtain vectors $\mathbf{A}_i = [A_{i1} \ldots A_{iJ_i}]$ and $\mathbf{B}_i' = [B_{i1}' \ldots B_{iJ_i}']$. Notice that $A_{ij} + B_{ij}' = l_{ij}(\mathbf{x})$.
   (b) Bob forms the vector $\mathbf{B}_i = [B_{i1} \ldots B_{iJ_i}]$, where $B_{ij} = B_{ij}' + \ln\alpha_{ij}$.
   (c) Alice and Bob engage in the secure maximum value protocol with vectors $\mathbf{A}_i$ and $\mathbf{B}_i$ to obtain y_i and z_i i.e. $y_i + z_i = SVAL(\mathbf{A}_i, \mathbf{B}_i)$.
   (d) Alice and Bob compute vectors $\bar{\mathbf{A}}_i = [(A_{i1} - y_i) \ldots (A_{iJ_i} - y_i)]$ and $\bar{\mathbf{B}}_i = [(B_{i1} - z_i) \ldots (B_{iJ_i} - z_i)]$.
   (e) Alice and Bob compute the dot product between the vectors $e^{\bar{\mathbf{A}}_i}$ and $e^{\bar{\mathbf{B}}_i}$ using $SIP(e^{\bar{\mathbf{A}}_i}, e^{\bar{\mathbf{B}}_i})$ and Bob gets the result of the dot product. Let the result be ϕ_i.
2. Bob computes the vector $\mathbf{u} = [u_1 \ldots u_N]$ where $u_i = z_i + \ln\phi_i + \ln P(\omega_i)$. Alice computes the vector $\mathbf{v} = [v_1 \ldots v_N]$ where $v_i = y_i$.
3. Alice and Bob perform the secure maximum index protocol between vectors $\mathbf{u}$ and $\mathbf{v}$ and Alice obtains $I = SMAX(\mathbf{u}, \mathbf{v})$.

Correctness: If one follows the protocol carefully, it is easy to see that $u_i + v_i$ is equal to $g_i(\mathbf{x})$.
Efficiency: For a given i, there are $(J_i(d+2)+1)$ SIP calls and 1 $SVAL$ call. Hence, in all, there are $(d{+}2)\sum_{i=1}^{N} J_i{+}N$ SIP calls, N $SVAL$ calls and 1 $SMAX$ call.

Security: If Protocol 1 and the protocols for SIP, $SVAL$ and $SMAX$ are secure, the above protocol is secure.

Notice that in step 1e, Bob receives the entire result of the inner product operation. Though this reveals no information about $\mathbf{x}$ to Bob, we can easily modify the step so that Alice and Bob receive additive shares ϕ_{iA} and ϕ_{iB} such that $\phi_{iA} + \phi_{iB} = \phi_i$. In step 2, Bob can compute u_i as $z_i + \phi_{iB} + \ln P(\omega_i)$ and Alice can compute v_i as $y_i + \phi_{iA}$ and the protocol will still hold. In case Alice and Bob want to compute additive shares of the likelihood instead of the class label, they will have to do an additional step of computing the logsum of $\ln \phi_{iA}$ and $\ln \phi_{iB}$ to obtain $\bar{\phi}_{iA}$ and $\bar{\phi}_{iB}$. They can then compute u_i as $z_i + \bar{\phi}_{iB} + \ln P(\omega_i)$ and v_i as $y_i + \bar{\phi}_{iA}$ and engage in $SVAL$ to compute the likelihood.

5. RESULTS, CONCLUSIONS AND FUTURE WORK

To validate the secure model we ran a large scale experiment on audio from sports television programs. The desired task was to learn and classify six different types of audio classes within the data. We performed the task twice, once without the privacy constraints by direct training and classification from the data, and once with the secure classifiers. The entire process between Alice and Bob was simulated using a MATLAB implementation. As expected the results from the two experiments were identical. The computation overhead was significantly more in the secure method, however the experiments were done using naively coded models and dramatic speedups can be obtained with careful optimization work. The use of different primitives can also result into widely varying performance, communication and security levels and a description of these is a lengthy research project of its own and out of the scope of this paper.

Using the same process we have also implemented secure Hidden Markov Models however due to space limitations we reserve the presentation of details for a future communication. Likewise, various signal processing and classification algorithms can be described in terms of secure primitives and reformulated in a secure cooperative manner. We expect this to be a fruitful area of research in the future. In addition to secure formulations there is also work that can be done in developing better protocols for the primitives used (SIP, $SMAX$ and $SVAL$) and to increase their efficiency.

Modifying algorithms already in existence to deal with secure cooperative models can be taken advantage of using the approach we have described. It is our hope that this process can help promote a more open collaboration setting where parties can freely exchange data and algorithms without legal and privacy issues.

6. APPENDIX

The following protocol is based on homomorphic encryption and was proposed by [2]. **Inputs**: Private vectors $\mathbf{x}$ and $\mathbf{y}$ with Bob and Alice respectively.
Outputs: Shares a and b such that $a + b = \mathbf{x}^t\mathbf{y}$.

1. Setup phase. Bob:
   - generates a private and public key pair $(\mathtt{sk}, \mathtt{pk})$.
   - sends $\mathtt{pk}$ to Alice.
2. For $i \in \{1, \ldots, d\}$, Bob:
   - generates a random new string r_i.
   - sends $c_i = \mathtt{En}(\mathtt{pk}; x_i, r_i)$ to Alice.
3. Alice:
   - sets $z \leftarrow \prod_{i=1}^{d} c_i^{y_i}$.
   - generates a random plaintext b and a random nonce r'.
   - sends $z' = z \cdot \mathtt{En}(\mathtt{pk}; -b, r')$ to Bob.
4. Bob computes $a = \mathtt{De}(\mathtt{sk}; z') = \mathbf{x}^t\mathbf{y} - b$.

See [2] for a proof that the protocol is correct and secure.

7. REFERENCES

[1] Yan-Cheng Chang and Chi-Jen Lu, “Oblivious polynomial evaluation and oblivious neural learning,” in *Advances in Cryptology, Asiacrypt ’01*, 2001, vol. 2248 of *Lecture Notes in Computer Science*, pp. 369–384.

[2] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen, “On private scalar product computation for privacy-preserving data mining,” in *Intl. Conference on Information Security and Cryptology*, C. Park and S. Chee, Eds., 2004, vol. 2506 of *Lecture Notes in Computer Science*, pp. 104–120.

[3] Shai Avidan and Moshe Butman, “Private information classification,” Unpublished, 2005.

[4] W. Du and M. J. Atallah, “Privacy-preserving cooperative statistical analysis,” in *Proceedings of the 17th Annual Computer Security Applications Conference*, New Orleans, Louisiana, December 2001.

[5] A. C-C. Yao, “Protocols for secure computation,” in *Proc. of the 23rd IEEE Symposium on Foundations of Computer Science*, 1982, pp. 160–164.

[6] W. Du and Z. Zhan, “A practical approach to solve secure multi-party computation problems,” in *Proceedings of New Security Paradigms Workshop*, Virginia Beach, virginia, USA, September 23-26 2002.

[7] M. J. Atallah, F. Kerschbaum, and W. Du, “Secure and private sequence comparisons,” in *Proceedings of Workshop on Privacy in the Electronic Society*, Washington, DC, USA, October 2003.