EFFICIENT PRIVACY-PRESERVING CLASSIFICATION OF ECG SIGNALS

*Mauro Barni, Pierluigi Failla, Riccardo Lazzeretti**

University of Siena, Italy

Annika Paus, Ahmad-Reza Sadeghi, Thomas Schneider

Ruhr-University Bochum, Germany

Vladimir Kolesnikov

Bell Laboratories, USA

ABSTRACT

We describe a privacy-preserving system where a server can classify an ElectroCardioGram (ECG) signal without learning any information about the ECG signal and the client is prevented from gaining knowledge about the classification algorithm used by the server. The system relies on the concept of Linear Branching Programs (LBP) and a recently proposed cryptographic protocol for secure evaluation of private LBPs. We study the trade-off between signal representation accuracy and system complexity both from practical and theoretical perspective. As a result, the inputs to the system are represented with the minimum number of bits ensuring the same classification accuracy of a plain implementation. We show how the overall system complexity can be strongly reduced by modifying the original ECG classification algorithm. Two alternatives of the underlying cryptographic protocol are implemented and their corresponding complexities are analyzed to show suitability of our system in real-life applications for current and future security levels.

Index Terms— Secure signal processing, privacy preserving, secure two-party computation, ECG classification.

1. INTRODUCTION

Health-care industry is moving faster than ever towards technologies offering personalized online self-service, medical error reduction, customer data collection and more. Such technologies have the potentiality of revolutionizing the way medical data is managed, stored, processed, delivered and ubiquitously made available to millions of users throughout the world. However, respecting the privacy of customers is a central problem, since privacy concerns may impede, or at least slow down, the diffusion of new e-health services.

In this paper, we consider a scenario for a remote diagnosis service. This service offers the analysis of biomedical signals to provide a preliminary classification in a potentially untrusted scenario. Such a system may either be seen as a stand alone service or as a part of a complete e-health system where the service provider, in addition to offering a repository of personal medical data, allows to remotely process such data. In order to preserve the privacy of the users, the server should carry out its task without getting any knowledge about the private data provided by the users. At the same time, the service provider may not be willing to disclose the algorithm it is using to process the signals, since they represent the basis for the service it is providing and valuable intellectual property. In general, one can resort to generic secure two-party computation (2PC) protocols [1, 2] allowing two parties to compute the output of a public function $f(\cdot)$ on their respective private inputs. At the end of the protocol, the only information obtained by the parties is the output of the function $f(\cdot)$, but no additional information about the other party's input. Specifically, we consider a variant of the above, where the function $f(\cdot)$ itself has to be kept secret. While this can be reduced to secure evaluation of a public function using universal circuits [3], this generic transformation poses an enormous additional overhead on the protocols. In this work, we consider the privacy-preserving classification of ElectroCardioGram (ECG) signals. Classification of ECG signals has long been studied by the signal processing community, but not yet in the context of a privacy-preserving scenario we aim to tackle in this paper. In our research we considered the secure implementation of a recently proposed classification algorithm [4]. The contribution of our research is fourfold. First, by relying on a recently proposed protocol for secure evaluation of Linear Branching Programs [5], we present an efficient system for privacy-preserving classification of ECG signals. Second, we link the representation accuracy of the to-be-processed signals (i.e., the number of bits representing the signals) and hence the complexity of the system, to the classification accuracy. Third, we show how the overall complexity of the system can be drastically reduced by tailoring the ECG classification algorithm to the 2PC scenario. Fourth, we compare two implementations of the secure protocol with respect to different parameter sizes and security levels showing that our system can be used in practice.

The rest of the paper is organized as follows. In §2 the plain version of the ECG classifier is described. In §3 the LBP concept and the protocols for secure evaluation of private LBPs are summarized. §4 is devoted to the description of the privacy-preserving ECG classification algorithm and the accuracy analysis. Experimental results regarding complexity are discussed in §5 and some conclusions are drawn in §6.

2. ECG CLASSIFICATION IN THE PLAIN DOMAIN

In this section, we describe the architecture of the plain domain version of the ECG classifier. In our system we are interested in classifying each heart beat according to 6 classes: Normal Sinus Rhythm (NSR), Atrial Premature Contractions (APC), Premature Ventricular Contractions (PVC), Ventricular Fibrillation (VF), Ventricular Tachycardia (VT) and SupraVentricular Tachycardia (SVT). The classification algorithm we use is inspired by the work of D. Ge et al. [4, chapter 8]. The choice of the algorithm is justified first of all by the good classification accuracy it ensures, secondly because it fits well the requirements of a privacy preserving implementation, finally because of its generality. As both, AutoRegressive (AR) models and Quadratic Discriminant Functions (QDF) are often used in automatic medical diagnosis, the protocol described in this

*contact author: riccardo.lazzeretti@gmail.com

The work was partially supported by the European Union under FP6 project SPEED and by MIUR (Italian Ministry of Education and Research) under project 2007JXH7ET. T. Schneider was supported by the European Union under FP7 project CACE.

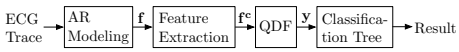

Fig. 1. Block diagram

paper may represent the basis for a large number of implementations addressing a variety of diverse topics in biomedical signal processing.

The overall architecture of the classifier is summarized by the block diagram in Figure 1. The input of the system is an ECG chunk corresponding to a single heart beat, that consequently, is classified as an independent entity. For the extraction of heart beats the algorithm proposed in [4] is used. We assume that the ECG signal is sampled at 250 sample per second and that 300 samples surrounding each peak are fed to the system: 100 samples preceding the beat peak and 200 following it. We also assume that the ECG signal has been pre-filtered by a notch filter removing the noise due to power line interference, electrode contact noise, motion artifact and base line wander [4].

The ECG classifier here taken into consideration relies on a rather general technique based on AR models for ECG description and a subsequent QDF classifier. Specifically, each ECG chunk is modeled by means of a 4-th order AR model. The AR model coefficients can be estimated in several ways; in our system we used a method based upon the Yule-Walker equations [6]. Once the AR model has been computed, five features[1] are extracted, yielding the following vector $\mathbf{f} = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, n_e)^T$. The first four features are the coefficients of the AR model and n_e is the number of samples for which the amplitude of the estimation error $|\epsilon_n|$ exceeds a threshold defined as $th = 0.25 \max_n (|\epsilon_n|)$. To perform a QDF classification as a linear operation, the classifier does not operate directly on $\mathbf{f}$. Instead a composite feature vector $\mathbf{f^c}$ is computed containing the features in $\mathbf{f}$, their square values and their cross products, namely:

$$\mathbf{f^c} = (1, f_1, \ldots, f_5, f_1^2, \ldots, f_5^2, f_1 f_2, \ldots f_4 f_5)^T = (f_1^c, \ldots f_{21}^c)^T.$$

The vector $\mathbf{f^c}$ represents the input of the QDF block in Figure 1. The QDF block projects $\mathbf{f^c}$ onto 6 directions $\boldsymbol{\beta}_i$, obtaining a 6-long vector $\mathbf{y}$, that represents the input of the final classification step: $\mathbf{y} = \mathbf{B}\mathbf{f^c}$, where $\mathbf{B}$ is a matrix whose rows are the vectors $\boldsymbol{\beta}_i$. The matrix $\mathbf{B}$ contains part of the knowledge embedded within the classification system, and is computed by relying on a set of training ECGs (see [4] for the details). For the final classification, the signs of the values y_i are extracted and used to actually classify the ECG, by means of the binary classification tree given in Figure 2.

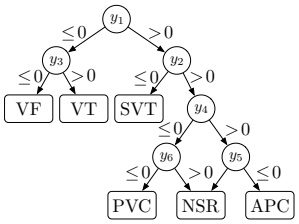

Fig. 2. Binary classification tree for ECG classification

3. SECURE EVALUATION OF PRIVATE LBPS

Our system is based on a recently proposed cryptographic protocol for efficient secure evaluation of private Linear Branching Programs (LBP) [5]. The notion of LBP is a natural generalization of binary classification trees and Ordered Binary Decision Diagrams (OBDDs). Compared to the above, LBPs have a more general branching condition that depends on the comparison of a linear combination of the inputs with a given threshold. More formally, LPB is defined as follows.

Definition 3.1 (Linear Branching Program (LBP) [5]) *Let* $\mathbf{x}^\ell = x_1^\ell, .., x_n^\ell$ *be the* **attribute vector** *consisting of signed* ℓ*-bit integer values. A binary* **Linear Branching Program (LBP)** $\mathcal{L}$ *is a triple* $\langle \{P_1, .., P_z\}, Left, Right \rangle$*. The first element is a set of* z *nodes consisting of* d **decision nodes** $P_1, .., P_d$ *followed by* $z - d$ **classification nodes** $P_{d+1}, .., P_z$*. Decision nodes* P_i*,* $1 \leq i \leq d$ *are the internal nodes of the LBP. Each* $P_i := \left\langle \mathbf{a_i^\ell}, t_i^{\ell'} \right\rangle$ *is a pair, where* $\mathbf{a_i^\ell} = \left\langle a_{i,1}^\ell, .., a_{i,n}^\ell \right\rangle$ *is the* **linear combination vector** *consisting of* n *signed* ℓ*-bit integer values and* $t_i^{\ell'}$ *is the signed* $\ell' = (2\ell + \lceil \log_2 n \rceil - 1)$*-bit integer* **threshold** *value with which* $\mathbf{a_i^\ell} \circ \mathbf{x}^\ell = \sum_{j=1}^n a_{i,j}^\ell x_j^\ell$ *is compared in this node.* $Left(i)$ *is the index of the next node if* $\mathbf{a_i^\ell} \circ \mathbf{x}^\ell \leq t_i^{\ell'}$*;* $Right(i)$ *is the index of the next node if* $\mathbf{a_i^\ell} \circ \mathbf{x}^\ell > t_i^{\ell'}$*. Functions* $Left(\cdot)$ *and* $Right(\cdot)$ *are such that the resulting directed graph is acyclic. Classification nodes* $P_j := \langle c_j \rangle$*,* $d < j \leq z$ *are the leaf nodes of the LBP consisting of a single classification label* c_j *each.*

Evaluation of the LBP $\mathcal{L}$ on an attribute vector $\mathbf{x}^\ell$ proceeds as follows. We start with the first decision node P_1. If $\mathbf{a_1^\ell} \circ \mathbf{x}^\ell \leq t_1^{\ell'}$, move to node $Left(1)$, else to $Right(1)$. Repeat this process recursively (with corresponding $\mathbf{a_i^\ell}$ and $t_i^{\ell'}$), until reaching one of the classification nodes and obtaining the classification $c = \mathcal{L}(\mathbf{x}^\ell)$.

3.1. Secure evaluation of private LBPs

The protocols for secure evaluation of private LBPs presented in [5] are executed between a server $\mathcal{S}$ and a client C. S has a private LBP $\mathcal{L}$, and $\mathcal{C}$ has the attribute vector $\mathbf{x}^\ell$. After protocol execution, $\mathcal{C}$ learns only the classification label $c = \mathcal{L}(\mathbf{x}^\ell)$ corresponding to his inputs. In particular, $\mathcal{C}$ learns nothing about $\mathcal{L}$ (besides the number of decision nodes d, and the length of the evaluation path e), and $\mathcal{S}$ learns nothing about $\mathcal{C}$'s inputs. For completeness, we include a general overview of the protocol. It consists of three main blocks (see Figure 3) as discussed below. The construction is somewhat similar to that of Yao's garbled circuit (GC) [1]. The main idea is to encrypt (or *garble*) the nodes and transitions of the LBP (algorithm CreateGarbledLBP), such that the evaluator of the garbled program (algorithm EvalGarbledLBP) is able to follow only a single evaluation path, defined by the LBP and the input attribute vector. The evaluation proceeds node by node, and the evaluator is able to decrypt and move to (only) the correct next node using the keys defined by CreateGarbledLBP and provided to C by protocol ObliviousLinearSelect. We now give the details of the three blocks.

CreateGarbledLBP. $\mathcal{S}$ creates a garbled version $\widetilde{\mathcal{L}}$ of the LBP $\mathcal{L}$. The garbled LBP $\widetilde{\mathcal{L}}$ maps the garbled inputs $\widetilde{w}_1, .., \widetilde{w}_d$ to the corresponding classification label c, and allows its oblivious evaluation. We note that for tiny LBPs, i.e., those having a small number of decision nodes d (which is the case in our application), this can be done most efficiently with a garbled Yao gate with d inputs [1].

ObliviousLinearSelect. In this phase, $\mathcal{C}$ obliviously obtains garbled values $\widetilde{w}_1, .., \widetilde{w}_d$, which correspond to the outcomes of the comparisons of the linear combination of the attribute vector with the threshold. $\mathcal{C}$ inputs the private attribute vector $\mathbf{x}^\ell = \{x_1^\ell, \ldots, x_n^\ell\}$,

[1] In [4, chapter 8] 6 features are used, however our experiments have shown that by using 5 features we obtain the same classification accuracy with a lower complexity.

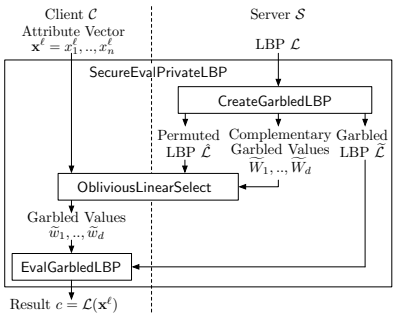

Fig. 3. Secure evaluation of private LBPs - protocol overview

and $\mathcal{S}$ inputs the private outputs of CreateGarbledLBP: complementary garbled values $\widetilde{W}_1 = \left\langle \widetilde{w}_1^0, \widetilde{w}_1^1 \right\rangle, .., \widetilde{W}_d = \left\langle \widetilde{w}_d^0, \widetilde{w}_d^1 \right\rangle$ and the permuted LBP $\hat{\mathcal{L}}$. (The latter consists of permuted linear combination vectors $\hat{\mathbf{a}}_\mathbf{1}^\ell, .., \hat{\mathbf{a}}_\mathbf{d}^\ell$ and permuted threshold values $\hat{t}_1^{\ell'}, .., \hat{t}_d^{\ell'}$.) Upon the completion of the protocol, $\mathcal{C}$ obtains the garbled values $\widetilde{w}_1, .., \widetilde{w}_d$ corresponding to the oblivious comparisons: if $\hat{\mathbf{a}}_\mathbf{i}^\ell \circ \mathbf{x}^\ell > \hat{t}_i^{\ell'}$, then $\widetilde{w}_i = \widetilde{w}_i^1$; else $\widetilde{w}_i = \widetilde{w}_i^0$. Two instantiations of ObliviousLinearSelect are given in [5].

Circuit instantiation. This instantiation, hereafter referred to as GC, securely evaluates a circuit which is naturally built from multiplication, addition and comparison circuits, as provided by [7].

Hybrid instantiation. In this instantiation, $\mathcal{C}$ encrypts each of its attributes with a semantically secure additively homomorphic encryption scheme (e.g., Paillier [8]) and sends these ciphertexts to $\mathcal{S}$. Using the additively homomorphic property, $\mathcal{S}$ can compute the linear combination under encryption, i.e., $Enc_{pk}(\hat{\mathbf{a}}_\mathbf{i}^\ell \circ \mathbf{x}^\ell)$. Finally, this ciphertext is obliviously compared with the threshold $\hat{t}_i^{\ell'}$ using a conversion protocol which combines homomorphic encryption and the evaluation of a small GC.

EvalGarbledLBP. The last phase is an analog of Yao's garbled circuit evaluation procedure. Here, $\mathcal{C}$ receives the garbled LBP $\widetilde{\mathcal{L}}$ from $\mathcal{S}$, and evaluates it on the garbled values $\widetilde{w}_1, .., \widetilde{w}_d$ output by ObliviousLinearSelect to obtain the classification label $c = \mathcal{L}(\mathbf{x}^\ell)$.

The protocols for secure evaluation of private LBPs are proven secure against semi-honest (or honest-but-curious) adversaries, and can be extended to be secure against malicious $\mathcal{C}$ as well [5].

4. PRIVACY-PRESERVING ECG CLASSIFICATION

Before describing the privacy-preserving ECG classification protocol, we define the players of the protocol and the data that needs to be protected. A first requirement is that the server $\mathcal{S}$, who is running the classification algorithm on client's ECG signal, learns neither any information about the ECG signal nor the final result of the classification. At the same time, the client $\mathcal{C}$ should not get any information about the algorithm used by $\mathcal{S}$, except for the output of the classification. The latter point deserves some explanation. We assume that the general form of the classifier used by $\mathcal{S}$ is known, however the parameters of the classifier need to be kept secret. By referring to the description given in §2, the algorithm parameters that $\mathcal{S}$ aims at keeping secret are the matrix $\mathbf{B}$ and the classification tree of Figure 2. This is a reasonable assumption since the domain specific knowledge needed to classify the ECGs and the knowledge got from the training, a knowledge that $\mathcal{S}$ may want to protect, reside in the classification tree and the matrix $\mathbf{B}$.

In order to introduce the privacy-preserving ECG classification protocol, we observe that the classification algorithm described in §2 is nothing but an LBP with $\mathbf{f}^{\mathbf{c},\ell}$ as attribute vector, and 6 nodes

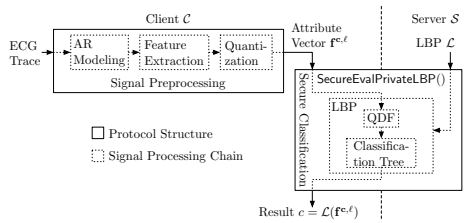

Fig. 4. Privacy-preserving ECG diagnosis

$P_i = \left\langle \boldsymbol{\beta}_\mathbf{i}^\boldsymbol{\ell}, 0 \right\rangle, i = 1, .., 6$, where $\mathbf{f}^{\mathbf{c},\ell}$ and $\boldsymbol{\beta}_\mathbf{i}^\boldsymbol{\ell}$ are ℓ-bit representations of the features and projection vectors. In this way, the general scheme for the privacy-preserving implementation of the classifier assumes the form given in Figure 4. All steps until the computation of the composite feature vector are performed by $\mathcal{C}$ on the plain data. Such a choice does not compromise the security of the system from the server's point of view, since $\mathcal{S}$ is not interested in keeping the structure of the classifier secret, but only in preventing users from knowing the matrix $\mathbf{B}$ and the classification tree. On the contrary, all the steps from the projection onto the directions $\boldsymbol{\beta}_i$'s, through the final classification are carried out securely. Note that with respect to the overall architecture depicted in Figure 1, we added a quantization step before the encryption of the composite feature vector. The need for such a step comes from the observation that the parameters $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ resulting from the AR model estimation procedure are usually represented as floating point numbers, a representation that is not suitable for 2PC protocols which compute on numbers represented as integers only. For this reason the elements of the composite feature vector $\mathbf{f}^\mathbf{c}$ are quantized and represented in integer arithmetic for subsequent processing[2]. Note that the choice of the quantization step, and consequently the number of bits used to represent the data (ℓ in the LBP terminology), is crucial since on one side it determines the complexity of the overall secure protocol and on the other side it has an impact on the accuracy of the ECG classification.

4.1. Quantization error analysis

In this section we estimate the impact that the quantization error introduced passing from $\mathbf{f}^\mathbf{c}$ to $\mathbf{f}^{\mathbf{c},\ell}$ and from $\boldsymbol{\beta}_\mathbf{i}$ to $\boldsymbol{\beta}_\mathbf{i}^\boldsymbol{\ell}$ has on the classification accuracy. Such an analysis will be used to determine the minimum number of bits (ℓ) needed to represent the attribute vector and the linear combination vectors of the LBP. The value of ℓ influences the complexity of the secure classification protocol for two main reasons. As already outlined in §3, the main ingredients of the protocols for secure evaluation of private LBPs are garbled circuits and additively homomorphic encryption. In the case of garbled circuits, the input of the protocol are the single bits used to represent $\mathbf{f}^{\mathbf{c},\ell}$ and $\boldsymbol{\beta}_\mathbf{i}^\boldsymbol{\ell}$. It is obvious, then, that the greater the number of bits, the more complex the resulting protocol will be. With regard to computing on homomorphically encrypted data, we observe that after each multiplication carried out in the encrypted domain, the number of bits necessary to represent the output of the multiplication increases (it approximately doubles)[3]. Since it is not possible to carry out truncations in the encrypted domain, it is necessary that the size of the ring used by the homomorphic cryptosystem is large enough to contain the output of the computations without an overflow which would cause an invalid result. Augmenting the number

[2] In the same way the coefficients of matrix B, representing the combination vectors of the LBP, are represented as integer numbers.

[3] The same observation holds for additions, however additions have a negligible effect with respect to multiplications.

of bits used to represent the inputs of the LBP may require to increase the size of the needed cryptosystem ring which results in an increased protocol complexity.

To start with, we observe that quantization is applied to the composite feature vector $\mathbf{f^c}$, that is used to compute the vector $\mathbf{y}$, through multiplication with the matrix $\mathbf{B}$. After such a step, only the signs of vector $\mathbf{y}$ are retained, hence it is sufficient to analyze the effect of quantization until the computation of the sign of $\mathbf{y}$. As to the processing steps carried out by the client prior to quantization, we assume that all the blocks until QDF are carried out by using a standard double precision floating point arithmetic. In order to simplify the notation, we consider only the computation of one coefficient of the vector $\mathbf{y}$. The function to be computed is a simple inner product: $y = \sum_j \beta_j f_j^c$ where the index i has been omitted, and β_j and f_j^c are real numbers. The quantized version of the above relationship can be expressed as follows:

$$\begin{aligned} \beta_{q,j} &= \rho_1 \beta_j + \varepsilon_{1,j} = \lfloor \rho_1 \beta_j \rceil \\ f_{q,j}^c &= \rho_2 f_j^c + \varepsilon_{2,j} = \lfloor \rho_2 f_j^c \rceil \end{aligned} \quad (1)$$

where ρ_1 and ρ_2 are positive integers and $\varepsilon_{1,j}$ and $\varepsilon_{2,j}$ are the quantization errors affecting $\beta_{q,j}$ and $f_{q,j}^c$ respectively. By using the above relations it is possible to evaluate the final error due to quantization:

$$\begin{aligned} &\sum_{j=0}^{N-1} \left(\rho_1 \beta_j + \varepsilon_{1,j}\right)\left(\rho_2 f_j^c + \varepsilon_{2,j}\right) = \\ &= \rho_1 \rho_2 \Bigg(y + \underbrace{\sum_{j=0}^{N-1} \frac{\beta_j \varepsilon_{2,j}}{\rho_2} + \sum_{j=0}^{N-1} \frac{f_j^c \varepsilon_{1,j}}{\rho_1} + \sum_{j=0}^{N-1} \frac{\varepsilon_{1,j}\varepsilon_{2,j}}{\rho_1 \rho_2}}_{\varepsilon} \Bigg) \end{aligned} \quad (2)$$

where ε indicates the error on the scalar product once the scaling factor $\rho_1\rho_2$ is canceled out. By letting $\max(|\beta_j|) = M_b$, $\max(|f_j^c|) = M_f$ and by noting that $\max(|\varepsilon_{1,j}|) = \max(|\varepsilon_{2,j}|) = \frac{1}{2}$, we have:

$$\varepsilon \leq \frac{N}{2\rho_1\rho_2}\left(\rho_1 M_b + \rho_2 M_f + \frac{1}{2}\right) \leq \varepsilon^* \quad (3)$$

where ε^* is a target maximum error that we do not want to exceed. Given ε^*, choosing the optimum values of ρ_1 and ρ_2 is equivalent to a constrained minimization problem in which the function to be minimized is $\rho_1\rho_2$ (since this is equivalent to minimize the number of bits necessary to represent the output of the scalar product) and the constraint corresponds to equation (3), that is:

$$\rho_1 \geq \frac{N(2\rho_2 M_f + 1)}{4\rho_2\varepsilon^* - 2NM_b}. \quad (4)$$

To ensure that ρ_1 is a positive integer, we require $2\rho_2\varepsilon^* - NM_b > 0$, yielding the following minimization problem:

$$\min_{\rho_2 > \frac{NM_b}{2\varepsilon^*}} \rho_2 \frac{N(2\rho_2 M_f + 1)}{4\rho_2\varepsilon^* - 2NM_b}. \quad (5)$$

By solving (5) we obtain the solutions:

$$\rho_2 = \frac{1}{2M_f\varepsilon^*}\left(NM_bM_f + \sqrt{NM_bM_f(\varepsilon^* + NM_bM_f)}\right), \quad (6)$$

$$\rho_1 = \frac{1}{2M_b\varepsilon^*}\left(NM_bM_f + \sqrt{NM_bM_f(\varepsilon^* + NM_bM_f)}\right). \quad (7)$$

4.2. Speeding up the system

By referring to the analysis in the previous section, we must consider that in our case $N = 21$, however the values of M_b and M_f are not known. In fact, the coefficients of the AR model and matrix B are not bounded. However, considering that in practical applications AR model coefficients are rather small (lower than 10 for ECG signals) and observing that the 5-th component of the feature vector $\mathbf{f}$ can be at most 300, hence in $\mathbf{f^c}$ we surely have a component that is at most $9 \cdot 10^4$. We may then let M_f to be the closest power of 10, i.e., $M_f = 10^5$. At the same time, in our experiments we never observed a matrix B with coefficients larger than $M_b = 10^5$. Finally by examining the data of the ECG MIT Database[4] we found that $\varepsilon^* = 10^{-5}$ ensures a sufficient classification accuracy. By using these settings the bit size of the values in input turned out to be 56 bit. As to the ring size for homomorphic encryption we found that the ring size imposed by security standards, e.g., 1248 bits and more [9], is always sufficient to accommodate all the intermediate and final results of the computation.

The analysis reported above is mainly based on worst case assumptions. In practice, we may expect that the number of bits necessary for a good classification accuracy is lower than 56. To investigate this aspect, we implemented a simulator to exactly understand which is the minimum number of bits that can be used. The results we obtained by running the simulator on the MIT Database of ECG signals are shown in Figure 5. This figure shows the accuracy of the system as a function of ℓ. As we can see $\ell = 44$ is sufficient to guarantee the same performance of a non-quantized implementation.

In order to further speed up the system, we tested a version of the ECG classifier with a reduced number of features. Specifically, we reduced $\mathbf{f}$ by eliminating the feature n_e. In this way, we obtain a 15-coefficient $\mathbf{f^c}$. Obviously the reduction of the feature space gives also a reduction of the accuracy, but this reduction is quite negligible: our experiments, in fact, indicate that the accuracy decreases only from 88.57% to 86.30%. On the other hand, as it will be shown in the next section, by removing one feature we gain a lot from a complexity point of view. Such a gain is already visible in Figure 5, where we can see that with the reduced set of features a value of ℓ as low as 24 is enough to obtain the same performance of a non-quantized version of the classifier.

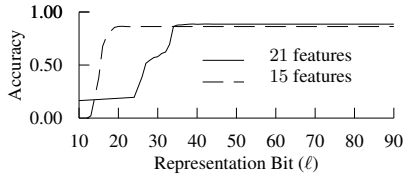

Fig. 5. Accuracy of dataset using 21 and 15 features

5. COMPLEXITY ANALYSIS

To evaluate the communication and computation complexity of the Hybrid and the GC protocols, we implemented both protocols in C++ using the Miracl library[5]. The following tests were run on two PCs with 3 GHz Intel Core Duo processor and 4GB memory connected via Gigabit Ethernet. The security parameters in the protocols of [5] are denoted by T for the bitlength of the RSA modulus for Paillier encryption [8] in the Hybrid protocol, and t for the symmetric security parameter which determines the performance of the GC

[4] http://www.physionet.org/physiobank/

[5] http://www.shamus.ie

protocol using an elliptic-curve based oblivious transfer protocol. In our implementation, we chose these security parameters according to common recommendations [9] as $T = 1248, t = 80$ for short-term security (recommended use up to 2010) and $T = 2432, t = 112$ for medium-term security (up to 2030). We measured the complexity of both protocol instantiations for the parameter sizes proposed in §4.2:

In test #1, we represent the features of $\mathbf{f}^{\mathbf{c},\ell}$ with $\ell = 56$ bits, as obtained from the theoretical estimations.

In test #2, the features are represented with $\ell = 44$ bits, the lower value obtained from the practical tests.

In test #3, we measure how the optimizations of §4.2 increase the efficiency of the protocols. While test-cases #1 and #2 were run for short-term security parameters only, in this test case we consider short-term (#3) and medium-term (#3*) security.

#	Features	N	ℓ	Protocol Type	Communication Client [kBytes] sent	Communication Client [kBytes] received	Computation Client [s] cpu	Computation Client [s] total	Computation Server [s] cpu	Computation Server [s] total
1	5	21	56	Hybrid	20.7	119.1	2.3	35.4	5.4	34.2
				GC	24.1	67435.6	7.2	64.5	17.3	64.7
2	5	21	44	Hybrid	17.7	94.5	2.0	29.0	4.8	27.6
				GC	19.0	41573.6	4.7	48.5	11.5	48.8
3	4	15	24	Hybrid	10.9	52.4	1.3	18.7	3.3	16.2
				GC	7.4	8788.4	1.3	17.5	3.1	19.2
3*	4	15	24	Hybrid	17.6	71.7	6.5	40.5	16.3	30.9
				GC	10.2	11984.3	3.0	20.4	4.6	20.8

* medium-term security

Table 1. Performance of protocols for secure ECG classification

Table 1 shows the results obtained from running these tests. Specifically, the table contains the communication complexity (separated into data sent and received by the client) and the computation complexity for the client and the server (separated into CPU time and total time which additionally includes data transfer and idle times). From these measurements we draw the following conclusions:

a) Parameter Sizes: The performance of both protocols in test #2 is slightly better than that of test #1 due to smaller size of ℓ. Reducing the number of features in test #3 results in substantially improved protocols while the classification accuracy is only slightly decreased as discussed in §4.2.

b) Communication Complexity: While the data sent by the client is approximately the same for both protocols (few kBytes), the received data in the GC protocol (MBytes) is by an order of magnitude larger than in the Hybrid protocol (kBytes). However, this asymmetric communication complexity of the GC protocol matches today's asymmetric network connections (e.g., ADSL or mobile networks), where the upstream is limited, while tens of MBytes can be downloaded easily. Future research should concentrate on further reducing the communication complexity of GC.

c) Computation Complexity (short-term security): For the test cases #1 and #2 the computation complexity of the Hybrid protocol is better by a factor of three in CPU time and factor two in total time, whereas for the optimized test case #3 both protocols have approximately the same computation complexity. Hence, for short-term security, the Hybrid protocol is better than the GC protocol with respect to computation and communication complexity (see also 'b)' above).

d) Computation Complexity (medium-term security): Increasing the security parameters has a much more dramatic effect on the computation complexity of the Hybrid protocol than on that of the GC protocol (see test #3 vs. #3*). This effect results from the asymmetric security parameter T being almost doubled, whereas the symmetric security parameter t is only slightly increased. We stress that this loss in performance of additively homomorphic encryption for realistic security parameter sizes is often neglected in literature or hidden by choosing relatively small moduli sizes of $T = 1024$ bit. For medium-term security, the GC protocol is substantially better than the Hybrid protocol besides the amount of data received by the client (see discussion in 'b)' above).

6. CONCLUSIONS

Privacy-preserving processing of medical signals calls for the application of cryptographic two-party computation techniques to medical signals. While in principle this is always possible, the development of efficient schemes that minimize the computation and communication complexity is not trivial, since it requires a joint design of the signal processing (SP) and cryptographic aspects of the system. In this paper we have presented an efficient and secure system for privacy-preserving classification of ECG signals based on a recently proposed 2PC protocol and a careful design of the SP algorithm used to classify the ECG. In particular, the optimization of the SP part substantially improved the performance of the secure protocols. We experimentally compared two different implementations of the system, one relying on garbled circuits (GC) and one on a hybrid combination of the homomorphic Paillier cryptosystem and GCs (Hybrid). While from communication complexity perspective the Hybrid protocol is clearly better, the computation complexity of both protocol is similar for short-term security parameters, whereas for medium-term security the GC based protocol is preferable.

7. REFERENCES

[1] A. C. Yao, "How to generate and exchange secrets," in *IEEE Symposium on Foundations of Computer Science (FOCS'86)*. 1986, pp. 162–167, IEEE.

[2] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella, "Fairplay — a secure two-party computation system," in *USENIX*, 2004, `http://www.cs.huji.ac.il/project/Fairplay`.

[3] V. Kolesnikov and T. Schneider, "A practical universal circuit construction and secure evaluation of private functions," in *Financial Cryptography and Data Security (FC'08)*. 2008, vol. 5143 of *LNCS*, pp. 83–97, Springer.

[4] U. R. Acharya, J. Suri, J. A. E. Spaan, and S. M. Krishnan, *Advances in Cardiac Signal Processing*, Springer, 2007.

[5] M. Barni, P. Failla, V. Kolesnikov, R. Lazzeretti, A.-R. Sadeghi, and T. Schneider, "Secure evaluation of private linear branching programs with medical applications," in *14th European Symposium on Research in Computer Security (ESORICS'09)*. 2009, LNCS, Springer, Full version available at `http://eprint.iacr.org/2009/195`.

[6] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time series analysis*, Holden-day San Francisco, 1976.

[7] A. Paus, A.-R. Sadeghi, and T. Schneider, "Practical secure evaluation of semi-private functions," in *Applied Cryptography and Network Security (ACNS'09)*. 2009, vol. 5536 of *LNCS*, pp. 89–106, Springer.

[8] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology – EUROCRYPT'99*. 1999, vol. 1592 of *LNCS*, pp. 223–238, Springer.

[9] D. Giry and J.-J. Quisquater, "Cryptographic key length recommendation," March 2009, `http://keylength.com`.