# ユーザの顔画像を秘匿可能な顔認証

西田 直央[1]　大庭 達海[1]　海上 勇二[1]　照屋 唯紀[2]　アッタラパドゥン ナッタポン[2]　花岡 悟一郎[2]

**概要**：顔画像認証は人の記憶力や物理媒体に依存しないため利便性が高く，昨今では決済時やアミューズメント施設における本人確認手段など多くの場面で利用されるようになってきた．認証される側の ID が必要となる 1 対 1 認証と異なり，ID を必要としない 1 対 N 認証は利便性がより高く，幅広い応用が期待される．一般にユーザが顔認証サービスを利用する際には登録時や利用時に顔画像をサービス提供者に渡さなければならないが，顔画像はプライバシ情報であるため，情報漏えい等を防ぐためにも秘匿されることが望ましい．本稿ではユーザの顔画像を秘匿したまま顔認証を行うシステムを提案する．提案手法では，Locality Sensitive Hashing(LSH) と秘匿計算を応用した 1 対 N の顔認証システムを考案し，ナイーブな全探索法と比較して高速であり，実用的な時間で動作可能なことを示す．

**キーワード**：秘密計算，秘密分散，顔認証，機械学習，LSH

# Face Recognition Using Secret Face Image

Naohisa Nishida[1]　Tatsumi Oba[1]　Yuji Unagami[1]　Tadanori Teruya[2]　Nuttapong Attrapadung[2]
Goichiro Hanaoka[2]

***Abstract:*** Face verification is highly convenient because it does not depend on human memory or physical media. Nowadays, it is increasingly used in many situations, such as at the time of payment and identity verification at amusement facilities. Unlike 1-to-1 authentication, which requires an authenticated ID, 1-to-N authentication that does not require an ID is more convenient and is expected to have a wide range of applications. In general, when a user uses the face verification service, The face image must be given to the service provider at the time of registration and verification. Since the face image is privacy information, it is desirable to keep it secret in order to prevent information leakage. In this paper, we propose a face verification system that keeps user face images secret. In the proposed method, a 1-to-N face recognition system using Locality Sensitive Hashing (LSH) and secret computation is devised, and it can be operated in a practical time.

***Keywords:*** Secure Computation, Secret Sharing, Face verification, Machine Learning, LSH

## 1. はじめに

深層ニューラルネットワークを用いた画像認識性能の向上に伴い，顔画像を利用した認証の精度が上がり，多くの場面で活用されるようになってきた．例えばコンビニエンスストアにおける決済や，アミューズメント施設やイベント会場への入場時の認証等に顔認証が用いられている事例がある．

顔認証には 1 対 1 認証と 1 対 N 認証の 2 種類が存在するが，本稿が対象とするのは 1 対 N 認証である．1 対 1 認証とは，照合する本人の ID を例えば ID カードなどから読み出し，その ID に紐づいて登録された顔画像と，撮影された顔画像の同一性を判断する方法である．一方 1 対 N 認証は，予め登録された複数の ID とその ID に対応する顔画像の中から，撮影された顔画像がどの ID の顔画像に対応するか，あるいはいずれの顔画像とも対応しないかを判

[1] パナソニック株式会社
Panasonic Corporation
[2] 国立研究開発法人 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology

定する方法である．

1 つの顔画像との同一性のみを判定すれば良い 1 対 1 認証と比較して，登録された全ての顔画像情報との比較を行う必要がある 1 対 N 認証の方が必然的に難易度が高くなるが，ID カード等の提示が不要であるため利便性に優れている．

1 対 N 認証を実現する方法として，例えば全ユーザの顔画像が大量に取得できるような場合は，機械学習を用いて直接的にクラス分類する方法が考えられる．しかしこの方法では，事前登録時にユーザが複数の顔画像を用意または撮影しなければならず，顔画像の登録に大きなコストがかかる．また，ユーザが増減するたびに分類モデルを学習しなおす必要があり，サービス提供者にとっても負担になる．そこで本稿では最近傍法を利用した 1 対 N 認証を考える．

1 対 N の顔認証を実現するシステムの例を次に示す．顔画像の登録フェーズでは，ユーザは 1 枚または少数の顔画像をサーバに送信し，サーバは深層ニューラルネットワークなどを用いて低次元の顔画像の特徴ベクトルを抽出し，ユーザと対応づけて保存する．認証フェーズでは，その場で撮影されたユーザの顔画像がサーバに送信される．サーバは送信された顔画像の特徴ベクトルの抽出を行い，最近傍点探索アルゴリズムを用いて，事前登録された全ユーザの顔画像の特徴ベクトルの中から，最も距離が近い特徴ベクトルを探し出し，その距離が所定の閾値未満の場合には認証成功とし，閾値を超える場合には認証失敗とする．

このようなシステムでは，ユーザが顔画像や，元の顔画像を推測可能な特徴ベクトルをサーバに登録する必要がある．しかし顔画像などの情報はユーザのプライバシー情報であるため，秘匿されることが望ましい．また，クラウドサーバ等でこれらの処理を行う場合，サービス提供者のノウハウをもとに作成された情報を利用する場合も多いため，顔認証システムで使用するパラメータ等も秘匿にすることが望ましい．本稿ではこの課題を解決するため，ユーザの入力やサービス提供者の情報を秘匿にしたまま顔認証を行うシステムを提案する．

本稿で想定するシナリオについて説明する．図 1 は想定するシナリオを表した図であり，顔認証を利用したイベント入退場サービスの提供者，秘匿顔認証を実行するクラウドサーバ，実際にイベントに参加するユーザを想定する．

サービス提供者は顔認証に必要な情報や学習モデル等を秘匿化してクラウドサーバへ格納する．ユーザはイベントに登録する際，自身の顔画像や顔画像の特徴量を秘匿化してクラウドサーバに送信する．そしてイベントの当日，ユーザはイベント受付で自身の顔写真を撮影し，秘匿化してクラウドサーバに送信する．クラウドサーバは，事前に登録された顔画像または顔画像の特徴量と送られたデータを比較し，マッチするユーザであれば参加を承認，なければ否認する．最終的に顔認証の結果はイベント受付に対して公開される．こうすることで，ユーザの顔画像に関する情報はユーザ本人以外には秘匿され，顔認証に必要な情報はサービス提供者以外に秘匿される．

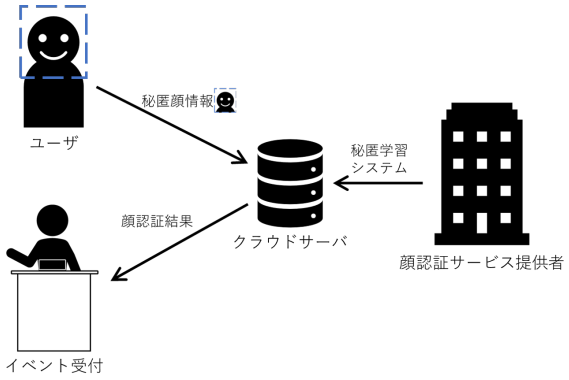


**図 1**　想定シナリオ

従来技術で上記のシナリオを実現する場合，クラウドサーバが平文状態でユーザの特徴量を保持する必要があったり，安全性を高めるために認証の精度を犠牲にする必要があった．

提案手法は，認証の精度を高めるため，深層ニューラルネットワークを用いた顔画像の特徴量抽出を前提とする．また，1 対 N 探索に必要となる最近傍探索法として，時間の掛かる全数探索を避け，秘密計算を用いた高速な最近傍探索に適した Locality Sensitive Hashing(LSH) を利用する．提案手法は秘密計算とハッシュテーブルの暗号化を用いることで高い安全性を達成しつつ，高速かつ高精度な顔認証システムを実現することを示す．

本稿の構成は次の通りである．2 章ではこの分野の関連研究を紹介する．3 章では提案手法のベースとなる平文での顔認証方式を説明し，4 章では提案手法で用いる秘密計算の説明を行う．そして 5 章では提案手法のキーアイデアと具体的な方式の構築を行い，6 章で実験結果を示す．7 章はまとめである．

## 2. 関連研究

本稿は，既存の顔認証アルゴリズムと秘匿計算技術をベースとした手法の構築を目指しているため，本節では顔認証と秘密計算技術に関する関連研究について示す．

### 2.1 顔認証の関連研究

深層ニューラルネットワークが大きく進展してきたことに伴い，顔認証に有用な，顔画像の低次元ベクトルへの埋め込みに関する研究が広く行われてきた [12][10]．これらの手法によって得られた低次元画像の埋め込みを用いることで，顔画像の類似度をコサイン距離やユークリッド距離で適切に得ることができるようになってきた．

### 2.2 最近傍探索の関連研究

最近傍探索法には厳密な最近傍を探索する手法と近似最近傍を探索する手法に大別され，更に近似最近傍探索法は Tree Partitioning や Graph 探索系，データをコンパクトなコードに変換する方法など様々な手法が提案されている [5]．中でも LSH[4][1] は古くから研究されてきた手法であり，多くの登録データからクエリされたデータとの最近傍を高速に探索するのに適している．

### 2.3 プライバシー保護近傍探索法の関連研究

ユーザの入力データを秘匿した状態で，効率的に最近傍のデータを応答するシステムが提案されている [15]．この手法はクエリされたデータのプライバシを守るのには適しているが，サーバが保持するデータベースは平文での保有を前提としており，情報漏洩に対する耐性は無い．一方，サーバ側では平文を保持せずに LSH の出力値のみを持たせ，かつ LSH の出力値のランダム性を高めることでサーバ側が保有するデータの秘匿性を担保しようとする手法がある [9]．しかしこの手法には安全性と利便性のトレードオフが存在し，安全性を高めると最近傍探索の精度が低下するという問題点がある．

## 3. 平文での顔認証システムの例

本章ではプライバシーを考慮しない 1 対 N の顔認証システムの一例を示す．ここで示すのは提案手法のベースとなる手法であり，顔画像からの特徴抽出は深層ニューラルネットワークによって行い，抽出された特徴量を LSH によって登録・認証するシステムである．提案手法がこの構成に基づく理由は 5.1 を参照．

### 3.1 深層ニューラルネットワークを用いた顔画像の特徴抽出

深層ニューラルネットワークを用いて顔認証に適した特徴ベクトルを取得する方法が提案されている [12][10]．これらの手法を用いることで，同一人物の顔画像から抽出された特徴ベクトルは，ユークリッド距離やコサイン距離の意味で近くなるという性質を持っている．

### 3.2 Locality Sensitive Hashing(LSH)

LSH は（特定の距離尺度で）近い入力値を同一のハッシュ値にマッピングさせる関数である．敢えて類似する入力値を衝突させるという意味で，暗号学的なハッシュ関数などとは大きく性質が異なる．LSH 関数族の中で今回利用するのは，2 つのベクトルのコサイン類似度が近い場合に同一のハッシュ値を返す SimHash と呼ばれる方法 [3] である．SimHash のアルゴリズムを Protocol.1，Protocol.2 に示す．入力ベクトルの次元を $d$ とすると，SimHash は $d$ 次元の単位ベクトルをランダムに生成し，入力ベクトルとランダムなベクトルの内積の符号を全ての単位ベクトルに対して算出し，1 つのベクトルとして出力する関数である．

**Protocol 1** SimHash Initialization

**Input:** $m$：出力ビット長
$d$：入力データ長
**Output:** $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m \in \mathbb{R}^d$：SimHash のパラメータ
**Procedure:**
1: **for** $i = 1$ to $m$ **do**
2: $\quad \boldsymbol{r}_i \leftarrow d$ 次元のランダムな単位ベクトル
3: **end for**
4: Output $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m$

**Protocol 2** SimHash

**Input:** $\boldsymbol{x} \in \mathbb{R}^d$：入力ベクトル
$\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m \in \mathbb{R}^d$：SimHash のパラメータ
**Output:** $\boldsymbol{y} \in \{0,1\}^m$：入力 $\boldsymbol{x}$ の SimHash 値
**Procedure:**
1: **for** $i = 1$ to $m$ **do**
2: $\quad s \leftarrow \mathit{InnerProduct}(\boldsymbol{x}, \boldsymbol{r}_i)$
3: $\quad t \leftarrow (s \geq 0)$
4: $\quad y_i \leftarrow t$
5: **end for**
6: Output $\boldsymbol{y}$

### 3.3 LSH を用いた顔認証プロトコル

1 対 N 認証を行う場合には，予め登録者の顔画像を全て特徴ベクトルに変換して保存し，認証の際には対象となる顔画像から抽出された特徴ベクトルに最も近い特徴ベクトルを見つけ出し，その特徴ベクトルとの距離が所定の閾値を下回るかどうかで人物の特定を行う．SimHash によるハッシュ値に基づく顔画像特徴量のハッシュテーブルを持つことで，入力された顔画像と類似する顔画像を効率的に探索することができる．

**Protocol 3** 顔登録

**Input:** $\mathrm{id} \in \mathbb{Z}$：登録されるユーザの ID
$\boldsymbol{x} \in \mathbb{R}^d$：顔画像の特徴量
$\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m \in \mathbb{R}^d$：SimHash のパラメータ
$\boldsymbol{T}$：ハッシュマップ（ハッシュ値 → 特徴量）
**Procedure:**
1: $\boldsymbol{y} \leftarrow \mathit{SimHash}(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_m)$
2: $\mathit{Append}(\boldsymbol{T}_{\boldsymbol{y}}, (\mathrm{id}, \boldsymbol{X}))$

## 4. 秘密計算

本章では一般的に知られている秘密計算のモデルを説明する．特に，単体のサーバが秘密計算するモデル，ユーザと単体もしくは複数のサーバが協力して秘密計算するモデル，複数のサーバが秘密計算するモデルの 3 つを想定する．

**Protocol 4** 顔認証

**Input:** $\boldsymbol{x} \in \mathbb{R}^d$：顔画像の特徴量
$\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m \in \mathbb{R}^d$：SimHash のパラメータ
$\boldsymbol{T}$：ハッシュマップ（ハッシュ値 → 特徴量)
$c$：閾値
**Output:** 認証結果
**Procedure:**
1: $\boldsymbol{y} \leftarrow LSH(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_m)$
2: $(\mathrm{id'}_1, \ldots, \mathrm{id'}_l, \mathrm{x'}_1, \ldots, \mathrm{x'}_l) \leftarrow \mathrm{T_y}$
($l$ はハッシュ値 $y$ に対応するユーザの数）
3: **for** $i = 1$ to $l$ **do**
4: $z \leftarrow \boldsymbol{x}$ と $\boldsymbol{x}'_i$ の類似度
5: **if** $z < c$ **then**
6: Output $\mathrm{id'}_i$
7: **end if**
8: **end for**
9: Output 否認

### 4.1 サーバ単体モデル

サーバ単体で秘密計算を行うモデルの例として準同型暗号（HE）が挙げられる．HE はユーザがユーザの鍵によってデータを暗号化し，サーバに預ける．サーバは暗号化データを復号することなく任意の計算を実行できる．

本稿で想定するシナリオでこのモデルを利用する場合，サービス提供者と全てのユーザが同じ鍵を共有しなければならない．多くのユーザが一つの鍵を共有するのは安全ではない．加えて，サービス提供者がユーザの情報を，ユーザがサービス提供者の情報を復号できてしまうため，本稿で想定するシナリオには向かない．

### 4.2 ユーザ-サーバモデル

ユーザとサーバが協力して秘密計算を行うモデルの例として 2 パーティ秘密計算が挙げられる．2 パーティ秘密計算はユーザとサーバが，互いに情報を秘匿にしたまま共有し，協力して計算を行うことで秘密計算を実行する．

このモデルの場合，ユーザの情報はサーバに対して秘匿できるが，一般的な方法の場合サーバはサービス提供者の情報を平文で持たなければならない．サービス提供者の情報を秘匿した状態でサーバ持つことが出来たとしても，スマホなどのユーザ端末が計算や相応のデータ通信しなければならないため，望ましいとは言えない．

### 4.3 複数サーバモデル

複数のサーバのみが協力して秘密計算を行うモデルの例としてマルチパーティ計算（MPC）が挙げられる．MPC は秘匿されたユーザ情報とサービス提供者の情報を用いて，それらを復号することなく任意の計算を行うことが出来る．

このモデルは，サーバ単体モデル，ユーザ-サーバモデルにあったような問題は発生しない．本稿ではこのモデルの秘密計算を想定する．

#### 4.3.1 秘密分散ベースの MPC

秘密分散法は，秘密情報を $n$ 個の値に分散して管理する手法である．この分散された値をシェアと呼ぶ．特に $(k, n)$ 閾値秘密分散法と呼ばれる手法は，シェアを $k$ 個以上集めることで秘密情報を復号することができるが，$k$ 個未満のシェアからは秘密に関する情報を一切得ることが出来ない．本稿では，秘密分散法の一つである，Shamir$(k, n)$ 閾値秘密分散法を取り上げる．

Shamir$(k, n)$ 閾値秘密分散法 [11] は，以下のアルゴリズムにより分散と復号を行う．文中の $p$ は秘密分散法の法，$\ell$ は法 $p$ のビット数である．

- 分散
  $s$ を秘密情報とする．$f(x) = s + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k-1} \, (mod\ p)$ $(\{a_1, \cdots, a_{k-1}\} \xleftarrow{\$} \mathbb{Z}_p^{k-1})$ を生成し，サーバ $i$ $(i = 1, 2, \cdots, n)$ に $[\![s]\!]_i = f(i)$ を送信する．
- 復号
  各サーバ $i$ は自分の持つシェア $[\![s]\!]_i$ を，他のサーバへ送信する．各サーバは，受け取った $(i, [\![s]\!]_i)$ を利用してラグランジュ補間公式（式 (1)）を用いて $f(0) = s$ を計算する．ここで $I$ は復号処理に参加するサーバの集合であり，$|I| \geq k$ ならば正しく復号ができる．

$$f(0) := \sum_{i \in I} [\![s]\!]_i \, \ell_i(0) \quad (I \subset \{1, \cdots, n\},\ |I| \geq k) \tag{1}$$

$$\ell_i(0) := \prod_{j \in I \setminus \{i\}} \frac{-j}{i - j}$$

Shamir$(k, n)$ 閾値秘密分散法は，MPC により秘密を復号することなく和や積を計算することが出来る．等号判定や大小比較等の上位プロトコルは和と積の組み合わせで実現可能であるため，MPC により計算可能である [13][2].

また，Shamir$(k, n)$ 閾値秘密分散法での MPC では，シェア同士の和，シェアと平文の和，シェアと平文の積を求める計算は，他のサーバと通信することなく実行できるが，シェア同士の積を計算するにはサーバ間の通信が必要であるため，実行により多くの時間を要する．上位プロトコルは前述のとおり，和積の組み合わせで構成可能であるため，計算量の評価としてシェア同士の積プロトコルの回数を用いる．この回数のことを通信量と呼ぶ．また，並列に積プロトコルを実行する際は，その通信を 1 回にまとめることができる．そのようにして通信回数を削減した際の最小通信回数をラウンド数と呼ぶ．

## 5. 提案手法

### 5.1 キーアイデア

提案手法は 3 章に記載したように，深層ニューラルネットワークによる顔画像の特徴抽出と，SimHash を用いた近似最近傍探索手法をベースとし，秘密計算技術を適用することで秘匿 1 対 N 顔認証の実現を目指す．

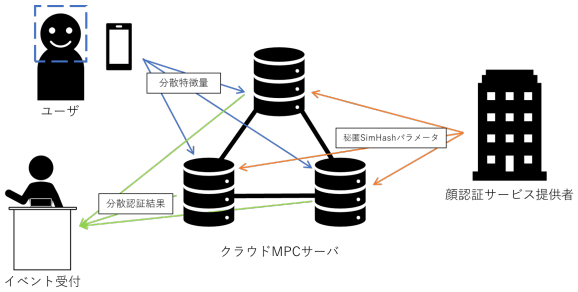


**図 2**　提案システム

Tree や Graph といった構造データを秘匿化したり秘匿状態で参照するのが困難なため，近傍探索手法としては Tree Partitioning や Graph 探索系の手法ではなく LSH やデータのコンパクト表現を得るための手法 [14][7] が適していると考えた．本稿ではデータが既に深層ニューラルネットワークによってコサイン距離の意味で類似度が良く表現され，適度に分散している低次元ベクトルが取得できているという前提に立つため，学習を必要とする手法ではなく，ランダムな射影を利用する LSH[4][1] を活用する．

ハッシュテーブルを複数作成することで最近傍探索の精度や速度の向上を図る手法も存在する [8]．この手法を用いた拡張を行うことも可能だが，本稿では最もシンプルなハッシュテーブルが 1 つの LSH(SimHash) を用いた手法を示す．

### 5.2　提案システム

本稿で提案するシステムを図 5.2 に示す．

#### 5.2.1　システムフロー

まず，事前にサービス提供者が行うフローを以下に示す．

(1) SimHash のパラメータを生成し，秘密分散して各サーバに配布する
(2) その他，秘匿学習システムに必要な情報やプログラムを各サーバに配布する
(3) ユーザに顔画像から特徴量を抽出するプログラムを配布する

顔画像から特徴量を抽出する方法にはいくつか種類があるか，本稿ではニューラルネットワークを用いる方法を想定する．学習モデルは，公開されている顔画像データセットと公開されているアーキテクチャを用いて学習した結果を用いるか，学習済みモデルには一般公開されているものもあるので，これを用いてもよい．どちらであろうと，この学習モデルにはサービス提供者のノウハウ等は含まれていないので，公開してもよい．

次に，イベント前およびイベント時にユーザとサーバが行う動作を以下に示す．

(1) ユーザは事前に自身の顔写真を撮影し，サービス提供者から与えられたプログラムで特徴量を抽出する
(2) ユーザは特徴量を秘密分散し，各サーバへ配布する
(3) イベント当日は，ユーザは自身の顔写真を撮影し，サービス提供者から与えられたプログラムで特徴量を抽出する
(4) ユーザは特徴量を秘密分散し，各サーバへ配布する
(5) 各サーバは得られた分散特徴量と，全ユーザが事前登録した分散特徴量を入力とし，秘匿顔認証を実行する
(6) 各サーバは結果として分散された顔認証結果を得る
(7) 各サーバはイベント受付に対して顔認証結果を送信し，イベント受付はそれを復号する

### 5.3　秘匿顔認証プロトコル

以下に，秘匿顔登録プロトコル（Protocol 6），秘匿顔認証プロトコル（Protocol 7）および秘匿 LSH プロトコル（Protocol 5）を示す．

**Protocol 5** 秘匿 SimHash

**Input:** $[\![\boldsymbol{X}]\!]\,(\boldsymbol{X} \in \mathbb{Z}_p^d)$：顔画像の秘匿特徴量
　$[\![\boldsymbol{r}]\!]\,(\boldsymbol{a} \in \mathbb{Z}_p^{m \times d})$：秘匿 SimHash パラメータ
**Output:** $[\![y]\!]\,(y \in \mathbb{Z}_p)$：入力 $\boldsymbol{X}$ の秘匿 SimHash 値
**Procedure:**
1: $[\![y]\!] \leftarrow 0$
2: **for** $i = 1$ to $m$ **do**
3:　$[\![s]\!] \leftarrow 0$
4:　**for** $j = 1$ to $d$ **do**
5:　$[\![s]\!] \leftarrow [\![s]\!] + \mathit{Mult}([\![\boldsymbol{X}_j]\!], [\![\boldsymbol{r}_{i,j}]\!])$
6:　**end for**
7:　$[\![t]\!] \leftarrow \left[\!\left[0 \le s \le \frac{p-1}{2}\right]\!\right]$
8:　$[\![y]\!] \leftarrow [\![y]\!] + \mathit{Mult}(2^i, [\![t]\!])$
9: **end for**

秘匿 SimHash プロトコルでは，特徴量が示す点が，SimHash パラメータで与えられた $m$ 個の平面よりも上にあるか下にあるかを判別する．そのため，特徴量 $\boldsymbol{X}$ とパラメータ $\boldsymbol{A}_i$ の内積にパラメータ $b$ を加え，その値が 0 以上か 0 未満かを計算している．ここで，法 $p$ が十分に大きいとき，0 以上 $\frac{p-1}{2}$ 以下の値は正の値を，$\frac{p+1}{2}$ 以上 $p-1$ 以下の値は負の値を示す．そのため本プロトコルでは，0 以上かの判定として，ステップ 8 のように，0 以上 $\frac{p-1}{2}$ 以下かを計算している．

**Protocol 6** 秘匿顔登録

**Input:** $[\![\mathrm{id}]\!] \in \mathbb{Z}_p$：登録されるユーザの秘匿 ID
　$[\![\boldsymbol{X}]\!]\,(\boldsymbol{X} \in \mathbb{Z}_p^d)$：顔画像の秘匿特徴量
　$[\![\boldsymbol{r}_1]\!], \ldots, [\![\boldsymbol{r}_m]\!]\,(\boldsymbol{r}_i \in \mathbb{Z}_p^d)$：秘匿 SimHash パラメータ
　$[\![\boldsymbol{T}]\!]$：秘匿ハッシュテーブル（ハッシュ値 → 秘匿特徴量）
**Procedure:**
1: $[\![y]\!] \leftarrow \mathit{SecureSimHash}([\![\boldsymbol{X}]\!], [\![\boldsymbol{r}_1]\!], \ldots, [\![\boldsymbol{r}_m]\!])$
2: $[\![h]\!] \leftarrow \mathit{SHash}([\![y]\!])$
3: $h \leftarrow \mathit{Reveal}([\![h]\!])$
4: $\mathit{Append}(\boldsymbol{T}_h, ([\![\mathrm{id}]\!], [\![\boldsymbol{X}]\!]))$

秘匿顔登録では，SimHash 値 $y$ のハッシュ値 $h$ を計算し，この $h$ をハッシュテーブルのインデックスとして使用

する．ハッシュ値から SimHash 値や顔情報を推測することは難しいため，このハッシュ値は公開することが出来る．秘匿顔登録では，SimHash 値 $y$ のハッシュ値 $h$ を計算し，この $h$ をハッシュテーブルのインデックスとして使用する．ハッシュ値から SimHash 値や顔情報を推測することは難しいため，このハッシュ値は公開することが出来る．この方式で用いる一方向関数は，出力の $h$ から SimHash 値 $y$ に関する情報が漏れず，同じ $y$ に対しては同じ $h$ を出力するものを採用する．また，より効率的にするために，衝突しない関数を採用することが望ましい．この条件を満たす一方向性関数としては，例えばあるランダムな値 $r$ を用いて，$SHash(\llbracket y \rrbracket, r) := \llbracket y^r \rrbracket[6]$ としてもいいし，AES を利用してもよい．

$y$ のとり得る値の範囲が狭い場合は，ハッシュ値 $h$ から SimHash 値 $y$ を総当たりで求められてしまう．そのため，秘匿乱数 $\llbracket r \rrbracket$ を用いるなどして $h$ にランダム性を持たせた方がよい．

**Protocol 7** 秘匿顔認証

**Input:** $\llbracket \boldsymbol{X} \rrbracket$ ($\boldsymbol{X} \in \mathbb{Z}_p^d$)：顔画像の秘匿特徴量
$\llbracket \boldsymbol{r}_1 \rrbracket, \ldots, \llbracket \boldsymbol{r}_m \rrbracket$ ($\boldsymbol{r}_i \in \mathbb{Z}_p^d$)：秘匿 SimHash パラメータ
$\boldsymbol{T}$：ハッシュテーブル（ハッシュ値 → 秘匿特徴量）
$SHash$：秘匿一方向関数 $c$：閾値
**Output:** $y$：秘匿認証結果
**Procedure:**
1: $\llbracket y \rrbracket \leftarrow SecureSimHash(\llbracket \boldsymbol{X} \rrbracket, \llbracket \boldsymbol{A} \rrbracket, \llbracket \boldsymbol{b} \rrbracket)$
2: $\llbracket h \rrbracket \leftarrow SHash(\llbracket y \rrbracket)$
3: $h \leftarrow Reveal(\llbracket h \rrbracket)$
4: $(\llbracket \mathrm{id'}_1 \rrbracket, \ldots, \llbracket \mathrm{id'}_l \rrbracket, \llbracket \boldsymbol{X'}_1 \rrbracket, \ldots, \llbracket \boldsymbol{X'}_l \rrbracket) \leftarrow \boldsymbol{T}_h$
($l$ はハッシュ値 $h$ に対応するユーザの数）
5: **for** $i = 1$ to $l$ **do**
6: $\quad \llbracket z \rrbracket \leftarrow \boldsymbol{X}$ と $\boldsymbol{X'}$ の類似度
7: $\quad \llbracket y \rrbracket \leftarrow \llbracket y \rrbracket + Mult(\llbracket \mathrm{id'}_i \rrbracket, \llbracket z < c \rrbracket)$
8: **end for**
9: Output $\llbracket y \rrbracket$

秘匿顔認証のステップ 7 ではテーブルに含まれている特徴量と，認証に用いる特徴量の類似度を計算している．この類似度は，ユークリッド距離でもいいし，コサイン類似度などを用いてもよい．

### 5.4 計算量

5.3 節で提案したプロトコルの計算量を表 1 に示す．なお，一方向性関数として公開されている乱数を冪数とする，べき乗関数 [6] を，類似度としてユークリッド距離を用いることとする．表中の記号は，$n$ が MPC におけるパーティ数，$\ell$ が秘密分散における法 $p$ のビット数，$m$ が LSH パラメータの次元数，$d$ が特徴量の次元数，$l$ は同じハッシュ値に対応するユーザの数を示す．

### 5.5 セキュリティモデル

閾値以上のサーバが結託しないことを仮定する．本稿で用いている MPC プロトコルは semi-honest モデルにおいて安全であるため，閾値以上のクラウドサーバが結託せず，クラウドサーバがプロトコルに従った動作をする限り，ユーザの顔画像はユーザ以外に秘匿され，LSH パラメータはサービス提供者以外に秘匿される．

**表 1** 計算量

| | ラウンド | 通信量 |
|---|---|---|
| SecureSimHash | 13 | $m*(93\ell+2)$ |
| 秘匿顔登録 | 15 | $m*(93\ell+2)+10n+1$ |
| 秘匿認証 | 22 | $m*(93\ell+2)+10n+1+(93\ell+2)*l$ |

**表 2** 速度評価

| $d$ | $m$ | $l$ | Offline 時間 (ms) | Online 時間 (ms) |
|---|---|---|---|---|
| 100 | 10 | 10 | 10.9 | 4.6 |
| 10000 | 10 | 10 | 22.0 | 17.3 |
| 10000 | 30 | 10 | 50.0 | 20.4 |
| 10000 | 30 | 30 | 57.3 | 45.7 |

## 6. 実験

実際に本提案方式を実装し，評価を行った．

### 6.1 セッティング

実装には C++，g++6.4.1 を用いた．実験では Amazon EC2 の c4.8xlarge インスタンスを用いた．同一リージョン内に 3 台のインスタンスを立ち上げることで，LAN 環境を疑似的に再現した．

MPC プロトコルには文献 [2] と文献 [13] に記載の方式がよく知られているが，文献 [2] の方式は大きな法 $p$ を用いなければならず通信量が膨大になるため，本稿では文献 [13] の方式を用いることとした．秘密分散には Shamir$(t, n)$ 秘密分散を採用し，閾値 $\boldsymbol{t = 2}$，パーティ数 $\boldsymbol{n = 3}$，法 $p = 2^{32} - 5$ として実験を行った．

実行時間については，乱数生成などの入力に依存せず，事前に計算できる処理にかかった時間を Offline 時間として，入力に依存する処理にかかった時間を Online 時間としてそれぞれ計測した．特に Online 時間は，実際のシステムを運用する際に重要になるため，Online 時間が少ない方が望ましい．

### 6.2 実験結果

LSH パラメータの次元数 $m$，特徴量の次元数 $d$，同じハッシュ値に対応するユーザの数 $l$ を変化させて数パターン比較実験を行った．

いずれも数十 ms から 100ms 程度で実行できており，本提案方式が実用的な時間で実行可能であることがわかる．

## 7. まとめ

本稿では，ユーザの顔情報とサービス提供者の持つ認証

パラメータを秘匿したまま顔認証を行う方式を提案した．提案方式では，LSH 値をそのままハッシュテーブルのインデックスとするのではなく，LSH 値のハッシュ値をインデックスとすることで，ユーザの顔画像に関する情報や，LSH のパラメータに関する情報を秘匿にしたまま，少ない計算量で実行できた．

また，実際に提案したプロトコルを実装し，パラメータをいくつか変化させて評価を行った．その結果，特徴量の次元数を 10000，LSH パラメータの次元数を 30，対象ユーザに近い顔のユーザ数を 30 とした場合でも，Offline 時間が 57.3$ms$，Online 時間が 45.7$ms$ であり，実用的な時間で実行可能であることがわかった．本稿ではパラメータをいくつか設定したが，今後は実際に使われている値を用いて評価したい．

## 参考文献


[1] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, January 2008.

[2] Octavian Catrina and Sebastiaan de Hoogh. Improved primitives for secure multiparty integer computation. In *Proc. of SCN 2010*, volume 6280 of *LNCS*, pages 182–199. Springer, 2010.

[3] Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Thiryfourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 380–388, New York, NY, USA, 2002. ACM.

[4] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM.

[5] Yusuke Matsui. 近似最近傍探索の最前線. `https://speakerdeck.com/matsui_528/jin-si-zui-jin-bang-tan-suo-falsezui-qian-xian`, 2019.

[6] Chao Ning and Qiuliang Xu. Constant-rounds, linear multi-party computation for exponentiation and modulo reduction with perfect security. In *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 572–589. Springer, 2011.

[7] Mohammad Norouzi and David J. Fleet. Minimal loss hashing for compact binary codes. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 353–360, USA, 2011. Omnipress.

[8] Ali Punjani. Fast search in hamming space with multi-index hashing. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 3108–3115, Washington, DC, USA, 2012. IEEE Computer Society.

[9] M. Sadegh Riazi, Beidi Chen, Anshumali Shrivastava, Dan S. Wallach, and Farinaz Koushanfar. Sub-linear privacy-preserving search with untrusted server and semi-honest parties. *CoRR*, abs/1612.01835, 2016.

[10] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.

[11] Adi Shamir. How to share a secret. *Communication of the ACM*, 22(11):612–613, 1979.

[12] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[13] T.Nishide and K.Ohta. Multiparty computation for interval, equality, and comparison without bit-decomposition protocol. *Public Key Cryptography*, 207:343–360, 2007.

[14] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'08, pages 1753–1760, USA, 2008. Curran Associates Inc.

[15] Chao Yan, Xuening Chen, and Qinglei Kong. Lsh-based private data protection for service quality with big range in distributed educational service recommendations. *EURASIP J. Wireless Comm. and Networking*, 2019:92, 2019.