

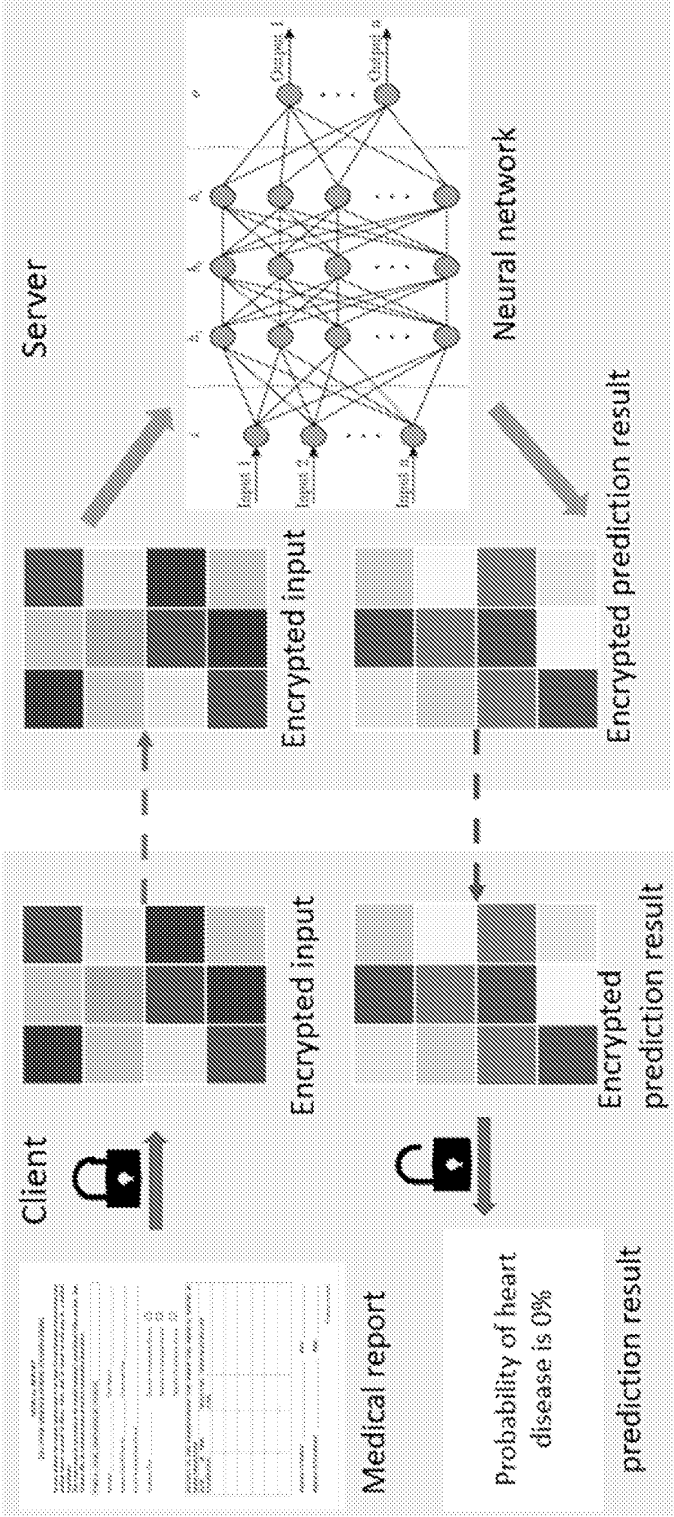


FIG. 1

2/28

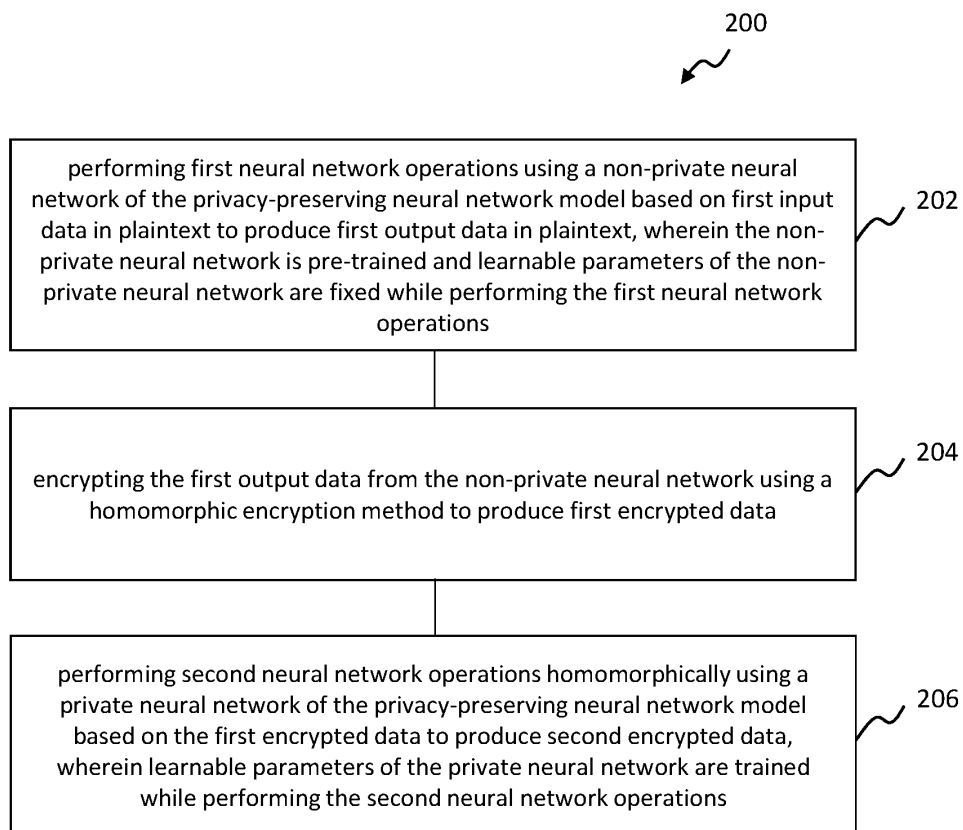


FIG. 2

3/28

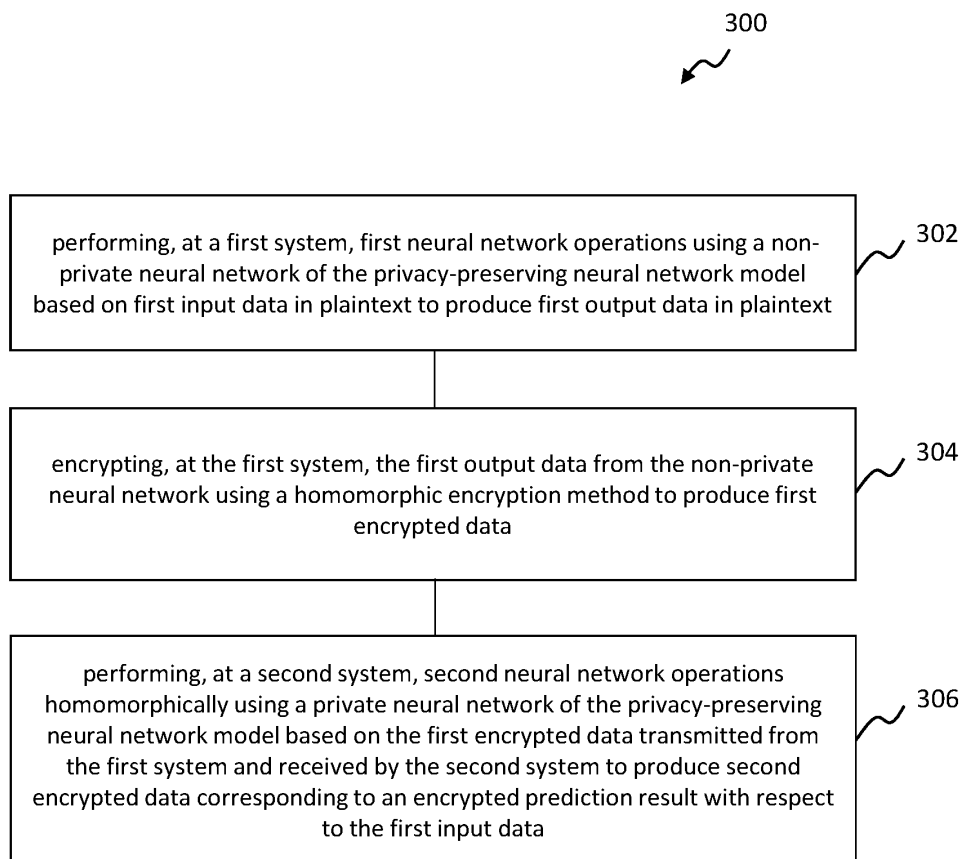


FIG. 3

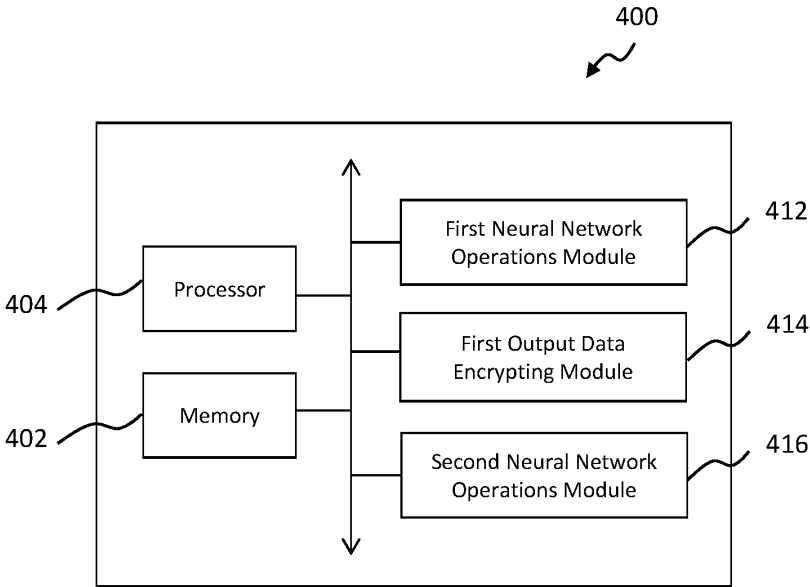


FIG. 4

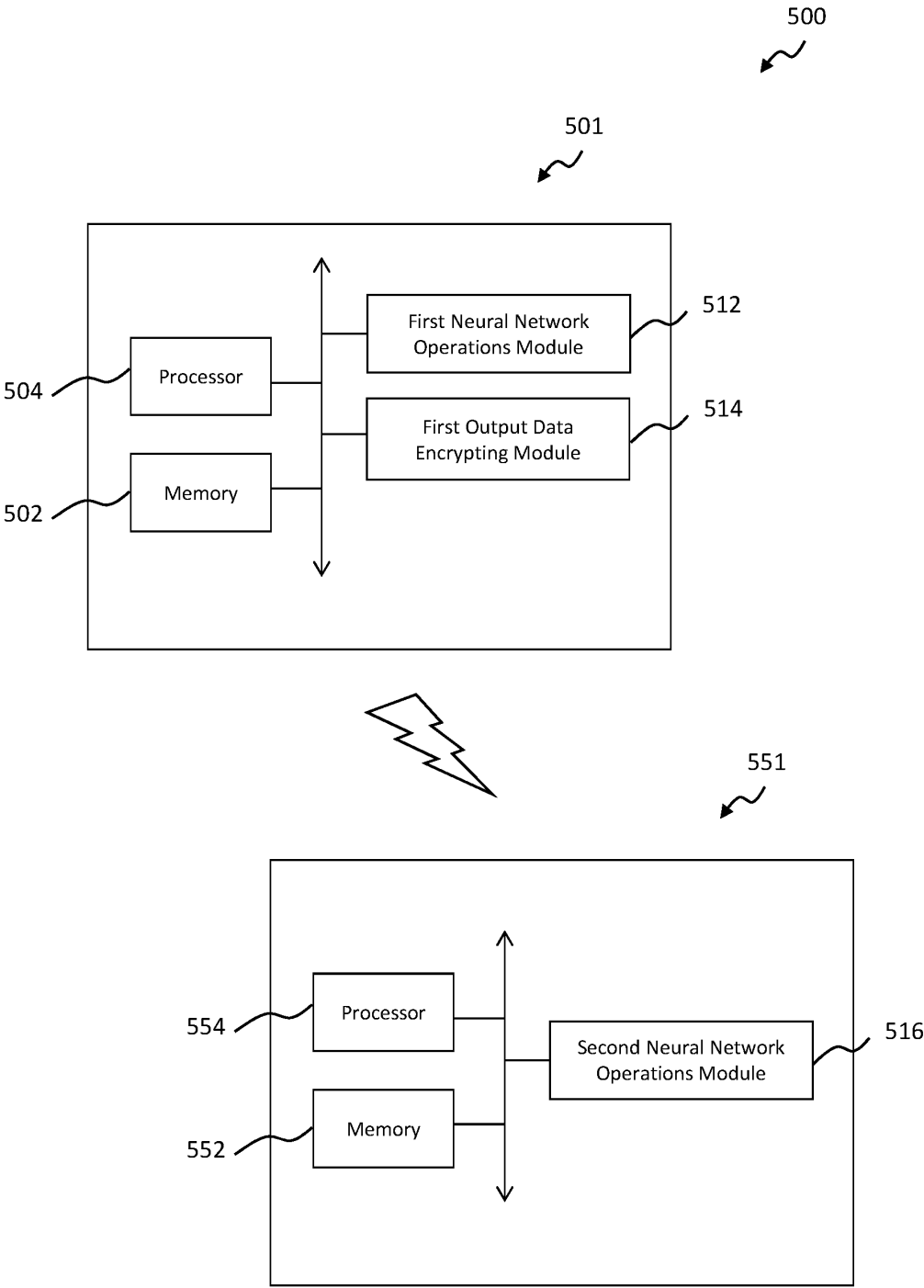


FIG. 5

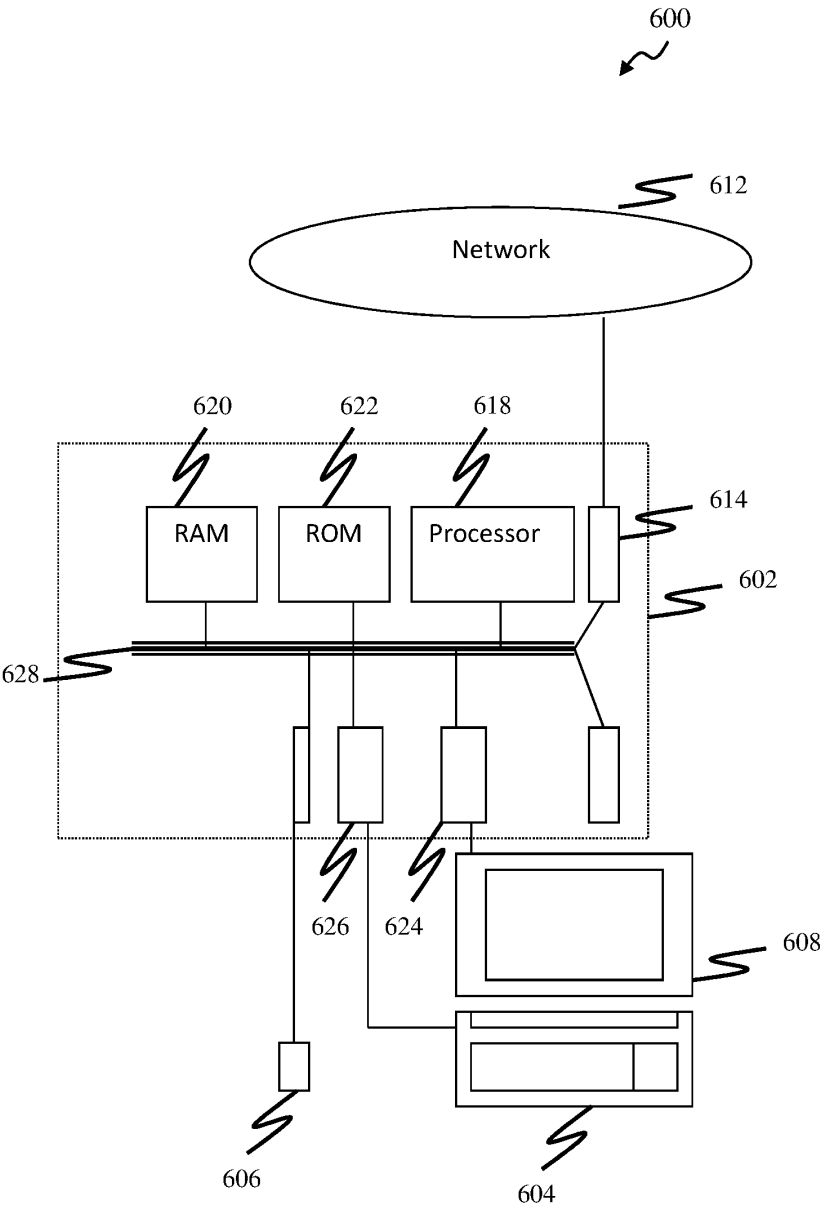


FIG. 6

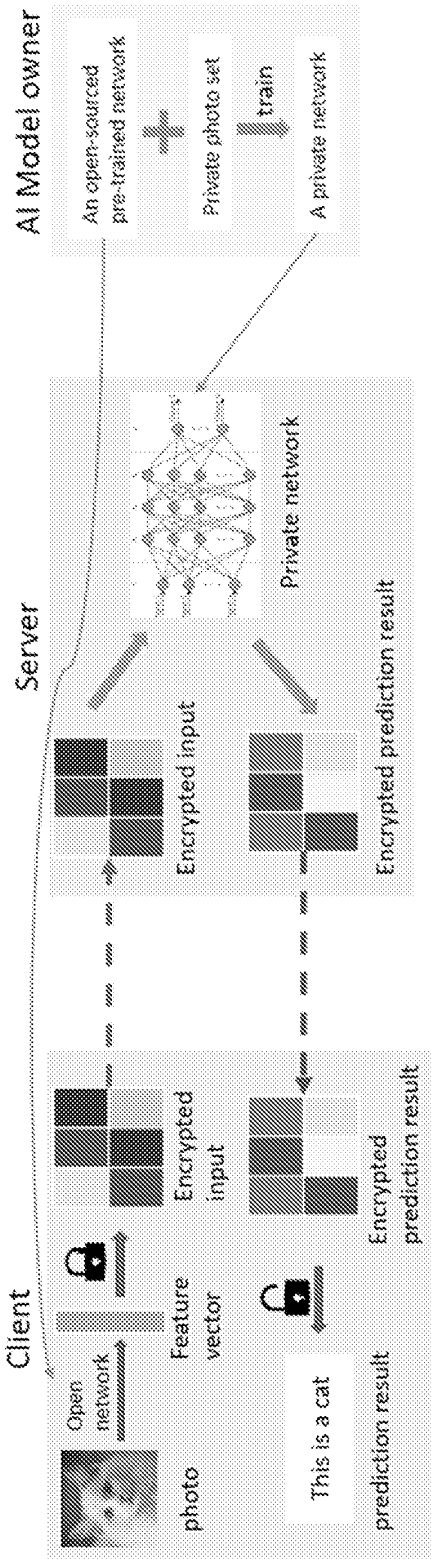


FIG. 7

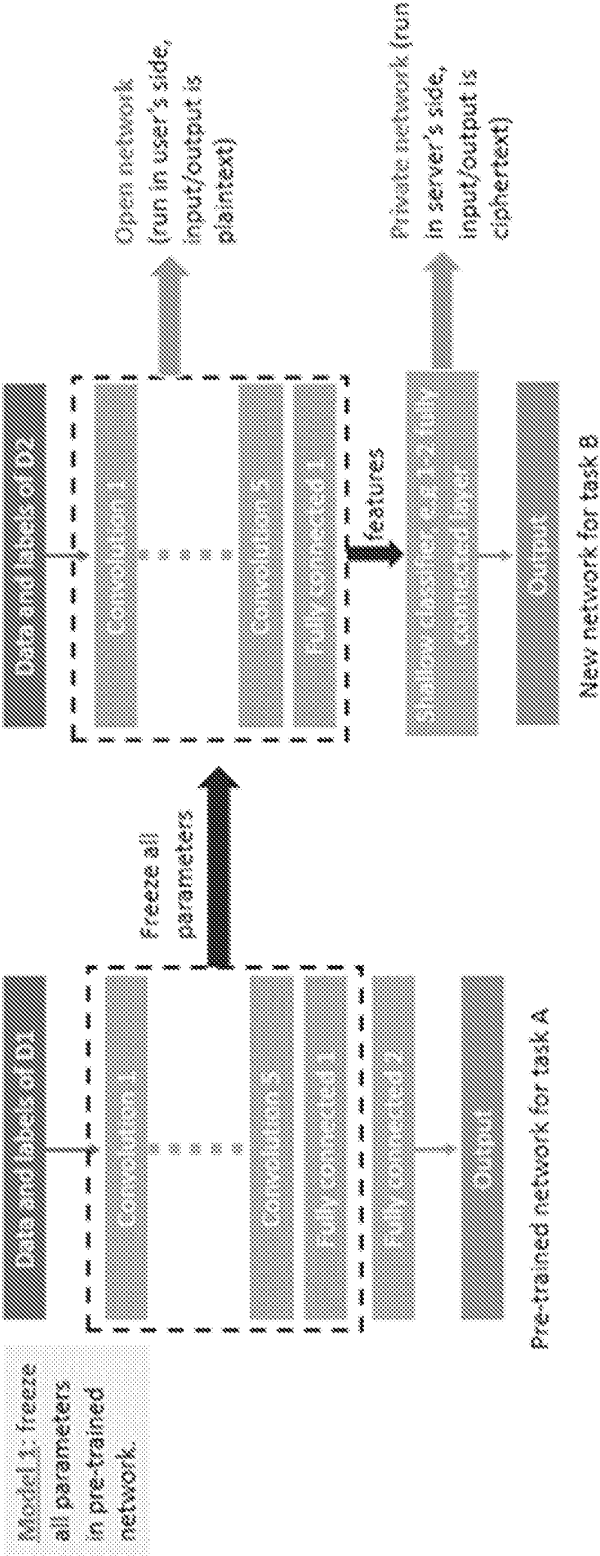


FIG. 8

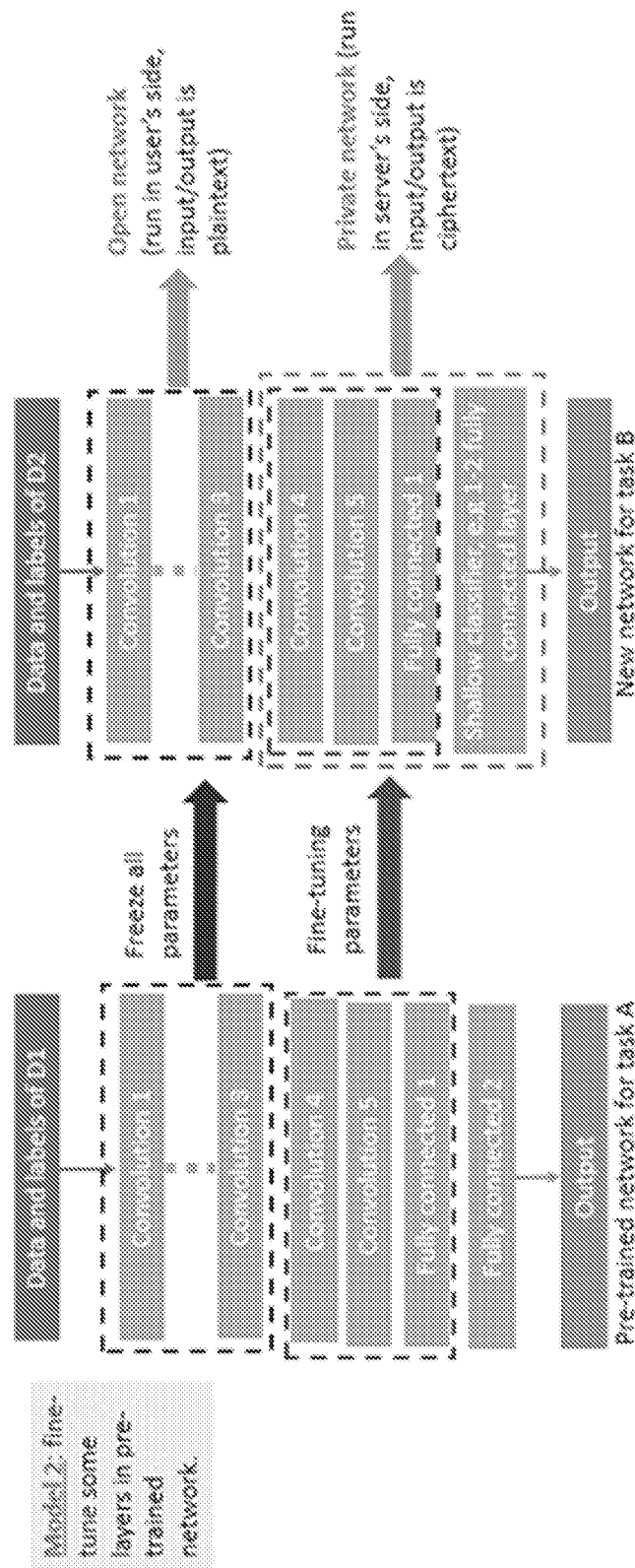


FIG. 9

10/28

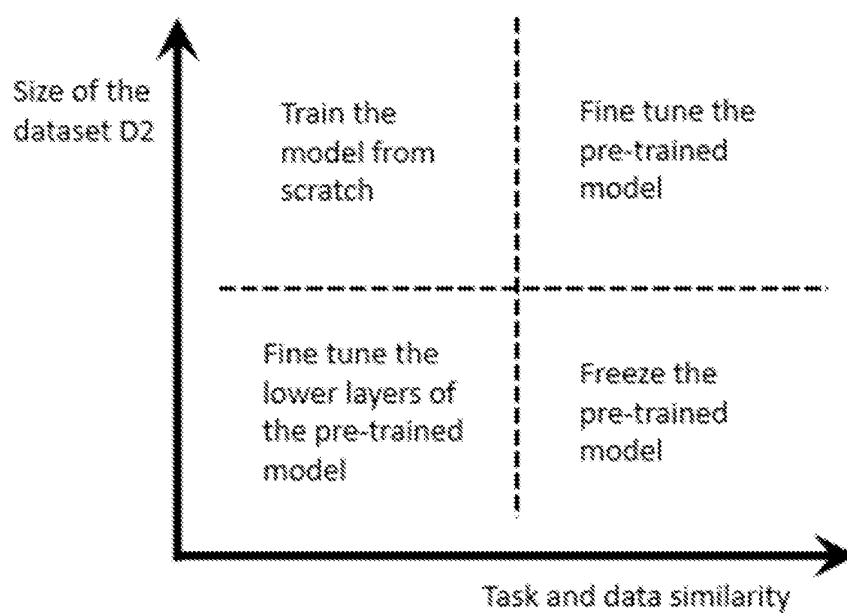


FIG. 10

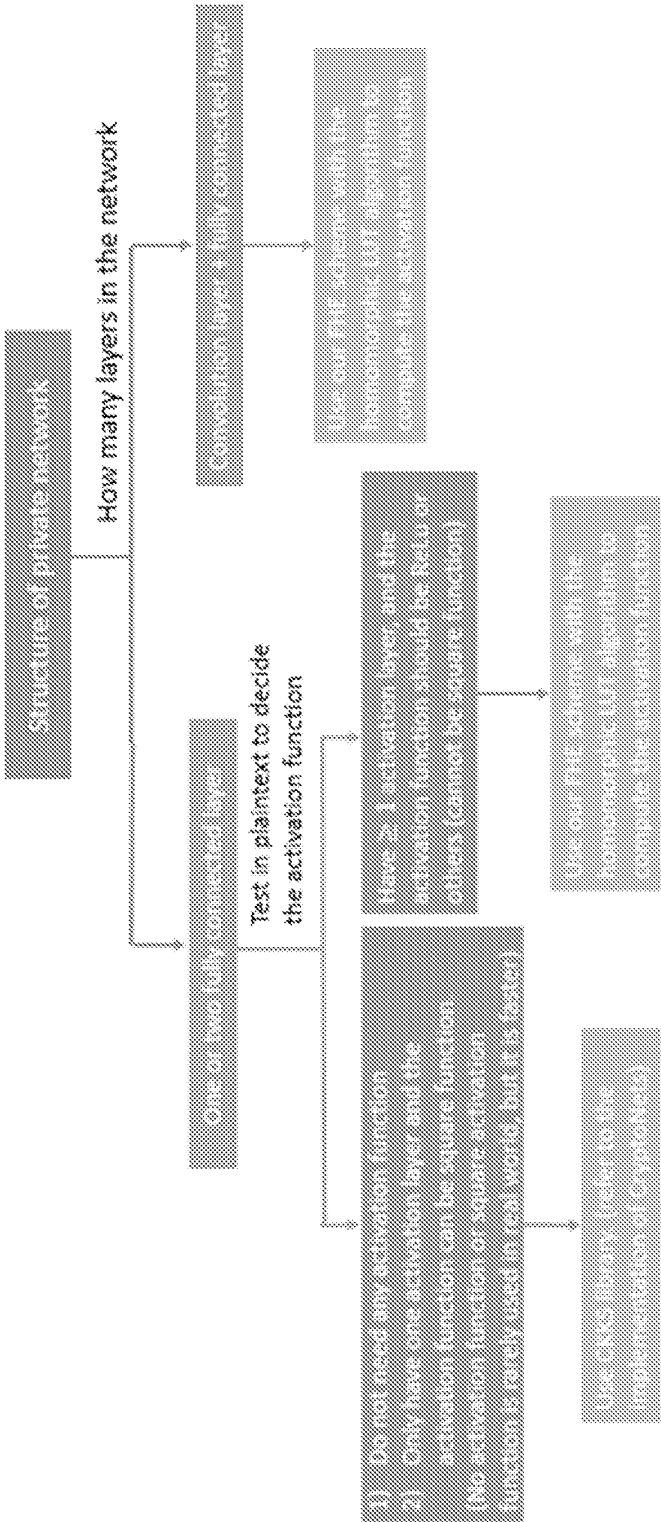


FIG. 11

12/28

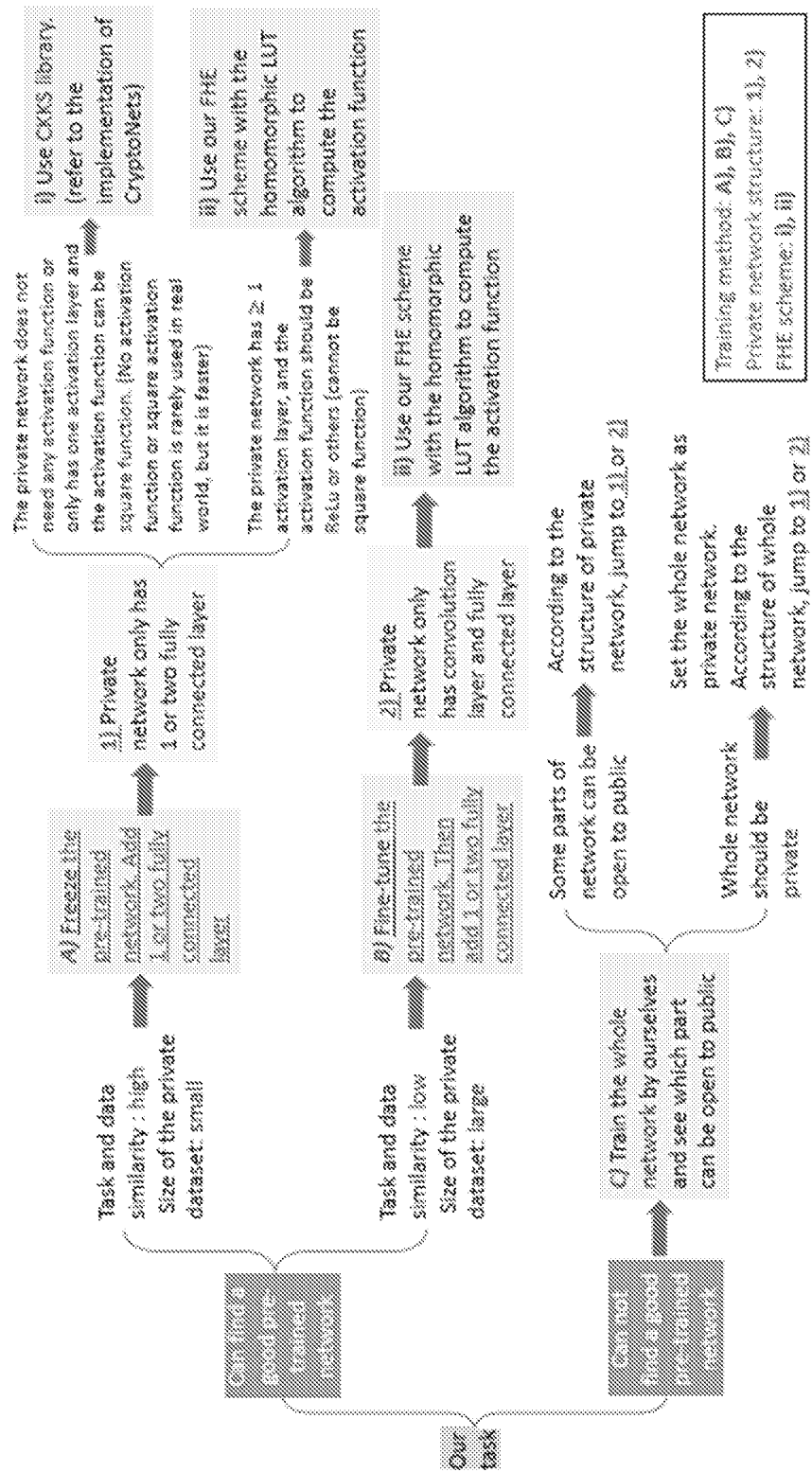


FIG. 12

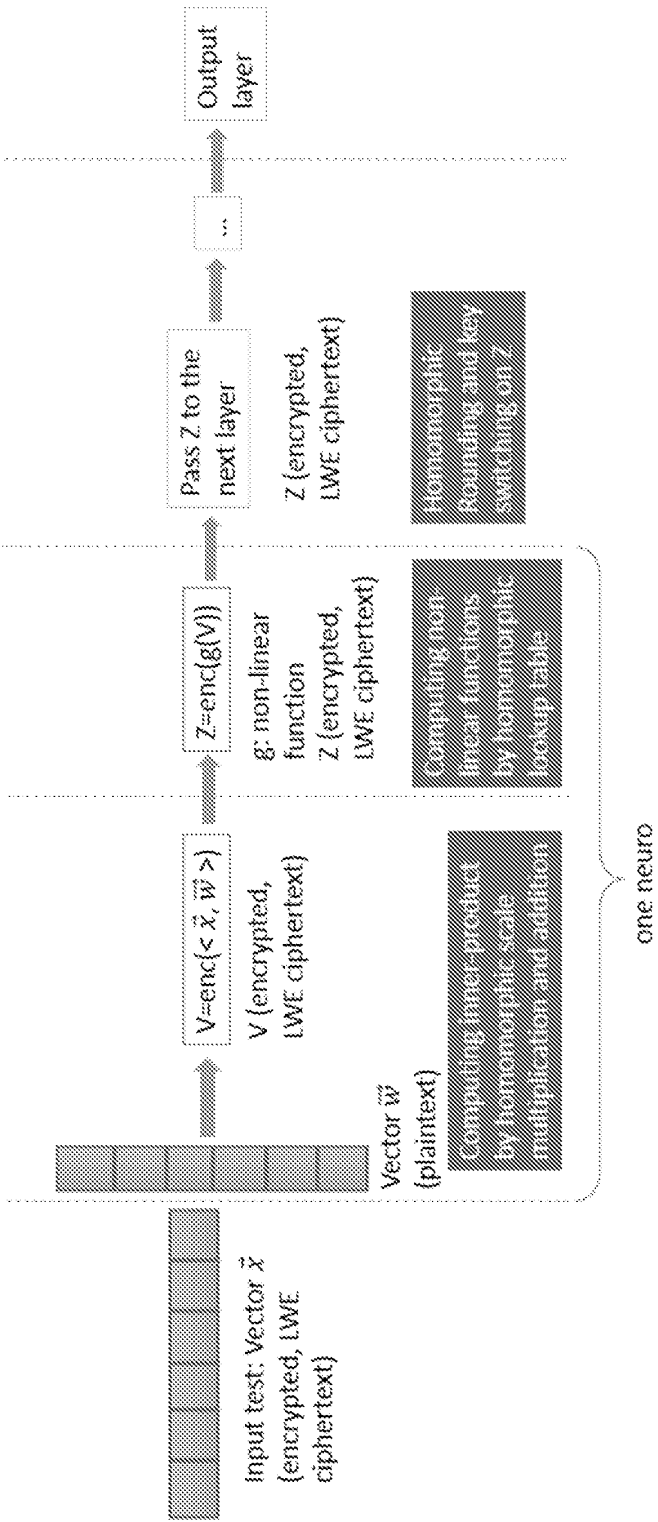


FIG. 13

14/28

Algorithm 1 LWE Encryption Scheme

Input: plaintext m , modulus q , dimension n , secret key $s \in \{-1, 0, 1\}^n$, error distribution $\mathcal{E}_{\text{LWE}}$.
Output: LWE ciphertext $(a, b) \in \text{LWE}_s^{n,q}(m)$.
 Sample error vector $e \in \mathcal{E}_{\text{LWE}}$.
 Sample vector $a \in \mathbb{Z}_q^n$.
 Define $b := m + e - \langle a, s \rangle \pmod q$.
 Return (a, b) .

FIG. 14

Algorithm 2 Scale multiplication

Input: LWE ciphertext $(a, b) \in \text{LWE}_s^{n,q}(m)$, plaintext $c \in \mathbb{Z}_q^n$.
Output: LWE ciphertext $(a', b') \in \text{LWE}_s^{n,q}(m)$.
 Compute $a' := c \cdot a \pmod q$.
 Compute $b' := c \cdot b \pmod q$.
 Return (a', b') .

FIG. 15

Algorithm 3 Addition

Input: LWE ciphertexts $(a_1, b_1) \in \text{LWE}_s^{n,q}(m)$ and $(a_2, b_2) \in \text{LWE}_s^{n,q}(m)$.
Output: LWE ciphertext $(a', b') \in \text{LWE}_s^{n,q}(m)$.
 Compute $a' := a_1 + a_2 \pmod q$.
 Compute $b' := b_1 + b_2 \pmod q$.
 Return (a', b') .

FIG. 16

Algorithm 4 RLWE Encryption Scheme

Input: plaintext m , modulus q , dimension n , secret key s whose coefficients are in $\{-1, 0, 1\}$, error distribution $\mathcal{E}_{\text{RLWE}}$.
Output: RLWE ciphertext $(a, b) \in \text{RLWE}_s^{n,q}(m)$.
 Sample error element $e \in \mathcal{E}_{\text{RLWE}}$.
 Sample element $a \in R_q^n$.
 Define $b := m + e - a \times s \in R_q^n$.
 Return (a, b) .

FIG. 17

Algorithm 5 Extended RLWE Encryption

Input: plaintext m , modulus q , dimension n , secret key $s \in \{-1, 0, 1\}^n$, error distribution $\mathcal{E}_{RLWE}$, decomposition base B .

Output: Extended RLWE ciphertext $\{(a_i, b_i)\}_{i \in [0, 1, \dots, \log(q)/\log(B)-1]} \in \text{RLWE}_s^{n,q}(m)$.

for $i = 0, 1, \dots, \log(q)/\log(B) - 1$ do

 Find (a_i, b_i) from $\text{RLWE}_s^{n,q}(m \times B^i)$.

end for

Return $\{(a_i, b_i)\}_{i \in [0, 1, \dots, \log(q)/\log(B)-1]}$.

FIG. 18

Algorithm 6 Extended RLWE Ciphertext Multiplication ($\odot$)

Input: plaintext operand r , modulus q , dimension n , decomposition base B , extended RLWE ciphertext $\{(a_i, b_i)\}_{i \in [0, 1, \dots, \log(q)/\log(B)-1]}$.

Output: RLWE ciphertext $(a, b) \in \text{RLWE}_s^{n,q}(r \times m)$.

Decompose r s.t. $r = \sum_{i=0}^{\log(q)/\log(B)-1} r_i \times B^i$.

Compute $(a, b) = \sum_{i=0}^{\log(q)/\log(B)-1} r_i \times (a_i, b_i) = \sum_{i=0}^{\log(q)/\log(B)-1} (r_i a_i, r_i b_i)$.

Return (a, b) .

FIG. 19

Algorithm 7 RGSW Encryption based on Extended RLWE Encryption

Input: plaintext m , modulus q , dimension n , secret key $s \in \{-1, 0, 1\}^n$.

Output: RGSW ciphertext $(\beta, \alpha) \in \left(\text{RLWE}_s^{n,q}(m), \text{RLWE}_s^{n,q}(s \times m) \right)$.

FIG. 20

Algorithm 8 RLWE and RGSW Multiplication ($\odot$)

Input: $(a, b) \in \text{RLWE}_s^{n,q}(m_1)$, $(\beta, \alpha) \in \text{RGSW}_s^{n,q}(m_2)$.

Output: $(a', b') \in \text{RLWE}_s^{n,q}(m_1 \times m_2)$.

Return $(a', b') = a \odot \alpha + b \odot \beta$.

FIG. 21

16/28

Algorithm 9 Extract0, to extract the constant term of an encrypted polynomial.

Input: $(a, b) \in \text{RLWE}_x^{n,q}(m)$.

Output: $(a', b') \in \text{LWE}_x^{n,q}(m(0))$ where s is the trivial vector form of polynomial s (from high degree to low degree).

 Let $s := s_{n-1}X^{n-1} + \dots + s_1X + s_0$, $a := a_{n-1}X^{n-1} + \dots + a_1X + a_0$ and $b := b_{n-1}X^{n-1} + \dots + b_1X + b_0$.

 Set $s := (s_{n-1}, \dots, s_1, s_0)$.

 Set $a' := (-a_1, \dots, -a_{n-1}, a_0)$ and $b' := b_0$.

 Return (a', b') .

FIG. 22

Algorithm 10 LWE Rounding and Rescale

Input: LWE ciphertext $(a, b) \in \text{LWE}_x^{n,q'}(m)$, another modulus q .

Output: LWE ciphertext $ct \in \text{LWE}_x^{n,q}(\lceil qm/q' \rceil)$.

 Return $(\lceil qa/q' \rceil, \lceil qb/q' \rceil)$.

FIG. 23

Algorithm 11 LWE Key Switch

Input: LWE ciphertext $(a, b) \in \text{LWE}_x^{n',q'}(m)$, another dimension n , another LWE secret key s , decomposition base B , a set of LWE switching keys w.r.t. B :

$$\text{SK}_j \in \text{RLWE}_x^{n,q} \left(\sum_{l=0}^{n-1} s[jn + l]X^l \right), j = 0, 1, \dots, n'/n - 1.$$

Output: LWE ciphertext $ct' \in \text{LWE}_x^{n,q}(m)$.

 Define a set of polynomials $\{\tilde{a}_j\}$ where

$$\tilde{a}_j := a[jn] - \sum_{l=1}^{n-1} a[jn + l]X^{n-l}, j = 0, 1, \dots, n'/n - 1.$$

 Compute a RLWE ciphertext $\tilde{ct} := \sum_{j=0}^{n'/n-1} \tilde{a}_j \circ \text{SK}_j$.

 Return $(0, b) + \text{Extract0}(\tilde{ct})$.

FIG. 24

17/28

Algorithm 12 Bit-by-bit Look-up Table Evaluation

Input: LWE ciphertext $(a, b) \in \text{LWE}_s^{n,q}(m)$ s.t. $|m| < q/4$, scale factor Δ , evaluation function $F(\cdot) : \mathbb{Z}_q \rightarrow \mathbb{Z}_q$, RLWE parameter set (n', q') , RLWE secret key s' , a set of evaluation keys w.r.t. the LWE secret key $s \in \{-1, 0, 1\}^n$:

$$\text{EK}_{j,0} = \text{RGSW}_{s'}^{n',q'}(1_{s[j] \geq 0}), \text{EK}_{j,n} = \text{RGSW}_{s'}^{n',q'}(1_{s[j] < 0}), j = 0, 1, \dots, n-1.$$

Output: LWE ciphertext $ct' \in \text{LWE}_{s'}^{n',q'}(\Delta F(m))$ where s' is the trivial vector form of polynomial s' (from high degree to low degree).

Let $\eta_k = kq/(2n')$ for $1 \leq k \leq n'/2$. Define a polynomial $f \in R_{n',q'}$ whose coefficients are:

$$f_j = \begin{cases} \lceil \Delta F(0) \rceil & \text{if } j = 0 \\ \lceil \Delta F(-\eta_j) \rceil & \text{if } 1 \leq j \leq n'/2 \\ \lceil -\Delta F(\eta_{n'-j}) \rceil & \text{if } n'/2 < j < n' \end{cases}.$$

Let $b' = \lceil 2n'b/q \rceil$, let $a' = \lceil 2n'a/q \rceil$ (slot-wisely).

Initialize $\text{AC} = (0, f \times X^{b'}) \in R_{n',q'}^2$.

for $j=0, 1, \dots, n-1$ do

$$\text{AC} += \left((X^{a'[j]} - 1)\text{EK}_{j,0} + (X^{-a'[j]} - 1)\text{EK}_{j,n} \right) \odot \text{AC}, \text{ note that all calculations are in } R_{n',q'}^2.$$

end for

Return $\text{Extract0}(\text{AC})$.

FIG. 25

Algorithm 13 2-bit Look-up Table Evaluation for single hidden layer

Input: LWE ciphertext $(a, b) \in \text{LWE}_s^{n,q}(m)$ s.t. $|m| < q/4$, scalar Δ , evaluation function $F(\cdot) : \mathbb{Z}_q \rightarrow \mathbb{Z}_q$, RLWE parameter set (n', q') , RLWE secret key s' , a set of evaluation keys w.r.t. the LWE secret key $s \in \{0, 1\}^n$:

$$\text{EK}_{j,0} = \text{RGSW}_{s'}^{n',q'}(s_{2j}s_{2j+1}), \text{EK}_{j,1} = \text{RGSW}_{s'}^{n',q'}(s_{2j}(1 - s_{2j+1})),$$

$$\text{EK}_{j,2} = \text{RGSW}_{s'}^{n',q'}(s_{2j+1}(1 - s_{2j})), j = 0, 1, \dots, n/2 - 1.$$

Output: LWE ciphertext $ct' \in \text{LWE}_{s'}^{n',q'}(\Delta F(m))$ where s' is the trivial vector form of polynomial s' (from high degree to low degree).

Let $\eta_k = kq/(2n')$ for $1 \leq k \leq n'/2$. Define a polynomial $f \in R_{n',q'}$ whose coefficients are:

$$f_j = \begin{cases} \lceil \Delta F(0) \rceil & \text{if } j = 0 \\ \lceil \Delta F(-\eta_j) \rceil & \text{if } 1 \leq j \leq n'/2 \\ \lceil -\Delta F(\eta_{n'-j}) \rceil & \text{if } n'/2 < j < n' \end{cases}.$$

Let $b' = \lceil 2n'b/q \rceil$, let $a' = \lceil 2n'a/q \rceil$ (slot-wisely).

Initialize $\text{AC} = (0, f \times X^{b'}) \in R_{n',q'}^2$.

for $j=0, 1, \dots, n/2-1$ do

$$\text{AC} += \left((X^{a'[2j]+a'[2j+1]} - 1)\text{EK}_{j,0} + (X^{a'[2j]} - 1)\text{EK}_{j,1} + (X^{a'[2j+1]} - 1)\text{EK}_{j,2} \right) \odot \text{AC}, \text{ all calculations are in } R_{n',q'}^2.$$

end for

Return $\text{Extract0}(\text{AC})$, i.e. the LWE ciphertext of the constant term in the plaintext of AC.

FIG. 26

Algorithm 14 2-bit Look-up Table Evaluation for multiple hidden layer

Input: LWE ciphertext $(a, b) \in \text{LWE}_n^{\Delta, \mathcal{A}}(m)$ s.t. $|m| < q/4$, scalar Δ , evaluation function $F(\cdot) : \mathbb{Z}_q \rightarrow \mathbb{Z}_q$, RLWE parameter set (n', q') , RLWE secret key s' , a set of evaluation keys w.r.t. the LWE secret key $s \in \{0, 1\}^n$:

$$\begin{aligned} \text{EK}_{j,0} &= \text{RGSW}_{s'}^{n', \mathcal{A}'}(1_{s[2j]=1, s[2j+1]=0}), \text{EK}_{j,1} = \text{RGSW}_{s'}^{n', \mathcal{A}'}(1_{s[2j]=1, s[2j+1]=1}), \text{EK}_{j,2} = \text{RGSW}_{s'}^{n', \mathcal{A}'}(1_{s[2j]=1, s[2j+1]=1, s[2j+2]=1}), \\ \text{EK}_{j,3} &= \text{RGSW}_{s'}^{n', \mathcal{A}'}(1_{s[2j]=0, s[2j+1]=1}), \text{EK}_{j,4} = \text{RGSW}_{s'}^{n', \mathcal{A}'}(1_{s[2j]=0, s[2j+1]=1, s[2j+2]=1}), \text{EK}_{j,5} = \text{RGSW}_{s'}^{n', \mathcal{A}'}(1_{s[2j]=0, s[2j+1]=1, s[2j+2]=1, s[2j+3]=1}), \\ \text{EK}_{j,6} &= \text{RGSW}_{s'}^{n', \mathcal{A}'}(1_{s[2j]=0, s[2j+1]=0}), \text{EK}_{j,7} = \text{RGSW}_{s'}^{n', \mathcal{A}'}(1_{s[2j]=0, s[2j+1]=0, s[2j+2]=1}), j = 0, 1, \dots, n/2 - 1. \end{aligned}$$

Output: LWE ciphertext $ct' \in \text{LWE}_n^{\Delta, \mathcal{A}'}(\Delta F(m))$ where s' is the trivial vector form of polynomial s' (from high degree to low degree).

Let $\eta_k = kq/(2n')$ for $1 \leq k \leq n'/2$. Define a polynomial $f \in R_{n', q'}$ whose coefficients are:

$$f_j = \begin{cases} [\Delta F(0)] & \text{if } j = 0 \\ [\Delta F(-\eta_j)] & \text{if } 1 \leq j \leq n'/2 \\ [-\Delta F(\eta_{n'-j})] & \text{if } n'/2 < j < n' \end{cases}.$$

Let $b' = \lceil 2n'b/q \rceil$, let $a' = \lceil 2n'a/q \rceil$ (slot-wisely).

Initialize $AC = (0, f \times X^{b'}) \in R_{n', q'}^{2, a'}$.

for $j=0, 1, \dots, n/2-1$ do

$$\begin{aligned} AC &+ = \left((X^{a'[2j]} - 1)\text{EK}_{j,0} + (X^{a'[2j]+a'[2j+1]} - 1)\text{EK}_{j,1} + (X^{a'[2j]+a'[2j+1]} - 1)\text{EK}_{j,2} \right. \\ &\quad + (X^{a'[2j+1]} - 1)\text{EK}_{j,3} + (X^{-a'[2j+1]} - 1)\text{EK}_{j,4} + (X^{-a'[2j]+a'[2j+1]} - 1)\text{EK}_{j,5} \\ &\quad \left. + (X^{-a'[2j]} - 1)\text{EK}_{j,6} + (X^{-a'[2j]+a'[2j+1]} - 1)\text{EK}_{j,7} \right) \odot AC, \text{ all calculations are in } R_{n', q'}^{2, a'}. \end{aligned}$$

end for

Return $\text{Extract0}(AC)$.

FIG. 27

Algorithm 15 Work Flow

Input Layer parameters: input LWE parameter set (n, q, s) .

Hidden Layer/Output Layer parameters: input LWE parameter set (n, q, s) , RLWE and RGSW related parameter set (n', q', s') where $n|n'$, look-up-table scalar Δ .

Public parameters: decomposition base B , Key-Switching decomposition base B_{KS} .

Given input vector M of length N , define a set of N LWE ciphertexts $\{(a_i, b_i)\}$ each of which is from $\text{LWE}_s^{n,q}(M[i])$.

Generate a set of evaluation keys w.r.t. the LWE secret key $s \in \{-1, 0, 1\}^n$:

$$\text{EK}_{j,+} = \text{RGSW}_{s'}^{n',q'}(\mathbf{1}_{s[j]>0}), \quad \text{EK}_{j,-} = \text{RGSW}_{s'}^{n',q'}(\mathbf{1}_{s[j]<0}), j = 0, 1, \dots, n-1.$$

Generate a set of LWE switching keys w.r.t. decomposition base B :

$$\text{SK}_j \in \text{RLWE}_s^{n,q}\left(\sum_{l=0}^{n-1} s[jn+l]X^l\right), j = 0, 1, \dots, n'/n-1.$$

Let input layer be Layer 0 and let output layer be Layer L .

for $l = 1, 2, \dots, L$ **do**

for $h = 1, 2, \dots, H_l$ **do**

$\text{LWE}_{l-1} := \{(a_i, b_i)\}$ each of which is from $\text{LWE}_s^{n,q}(\cdot)$ be the output from Layer $l-1$. For the sake of simplicity we omit the subscript of l in (a_i, b_i) .

 Find the inner-product input to the h -th node in Layer l : $ip_h = \sum_i w_l[i, h] \times (a_i, b_i) + b_l[i] \in \text{LWE}_s^{n,q}(\cdot)$.

 Evaluate the look-up-table in the h -th node: $ct'_h = \text{LUT}(ip_h) \in \text{LWE}_{s'}^{n',q'}(\Delta \times \cdot)$.

 Switch from n' to smaller n : $ct_h = \text{LWE_KS}(ct'_h, B_{KS}) \in \text{LWE}_s^{n,q'}(\Delta \times \cdot)$.

 Round from q' to smaller q and rescale: $ct_h = \text{LWE_Rounding}(ct_h) \in \text{LWE}_s^{n,q}(\cdot)$.

 Include ct_h in LWE_l .

end for

end for

Return LWE_L to User. User can decrypt all ciphertexts in it with secret key s , and then find the prediction result.

FIG. 28

Table 1: Comparison of LUT algorithms

	FHE-DINN	PEGASUS	Our 1-bit LUT	Our 2-bit LUT for single layer	Our 2-bit LUT for multiple layers
External products	$n/2$	$2n$	n	$n/2$	$n/2$
Polynomial multiplication	$2ln$	$8ln$	$4ln$	$2ln$	$2ln$
Quick multiplication	$12ln$	$4n$	$8ln$	$12ln$	$32ln$
RGSW ciphertext in EK	$3n/2$	$2n$	$2n$	$3n/2$	$4n$

FIG. 29

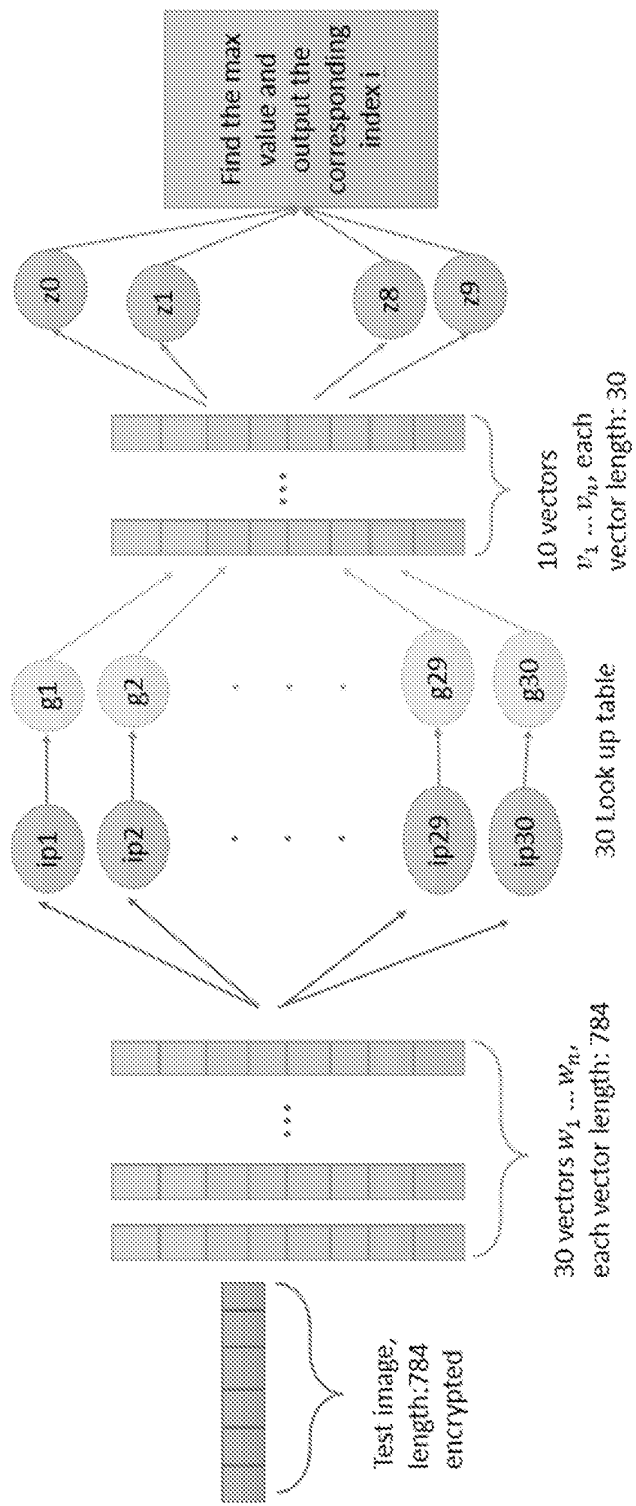


FIG. 30

Table 2: Evaluation results in BP network (1 hidden layer, 30/100 nodes)

BP Network (Activation function)	Number of nodes	Prediction accuracy in the plaintext	Homomorphic evaluation system	Time in lab PC	Prediction accuracy
Our model (ReLU)	30	94.80%	Ours	2.4s per test (4 threads)	94.04%
Our model (ReLU)	100	97.30%	Ours	7.3s per test (4 threads)	96.70%
Our model (ReLU)	30	94.80%	PEGASUS	100 min for 256 tests (4 threads) About 23s per test	91.39%
FHE-DiNN (sign function)	30	94.76%	FHE-DiNN	10.95s per test (1 thread)	93.71%

FIG. 31

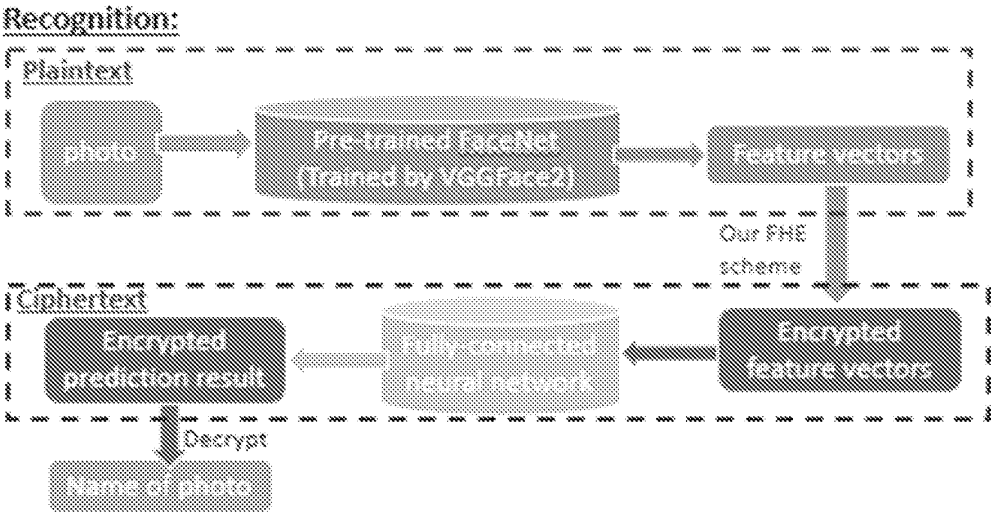


FIG. 32

Table 3: Evaluation results in face recognition

	Open network (run in lab PC)	Private network (run in Google Cloud)		Results	
Network model	Open network structure	Private network structure	Number of activation function	Time per recognition	Prediction accuracy
Our hybrid neural network	Deep convolution neural network	Simple fully connected network	30	2.1s	100%
Traditional neural network	Deep convolution neural network + simple fully connected network	-	0	10ms	100%
Basic privacy- preserving neural network	-	Deep convolution neural network + simple fully connected network	$\approx 6M$	≈ 5 Days (estimated)	-

FIG. 33

LUT+KS+Rounding Error of Multi-Layer
(Inner Product Range: ~1000~1000)

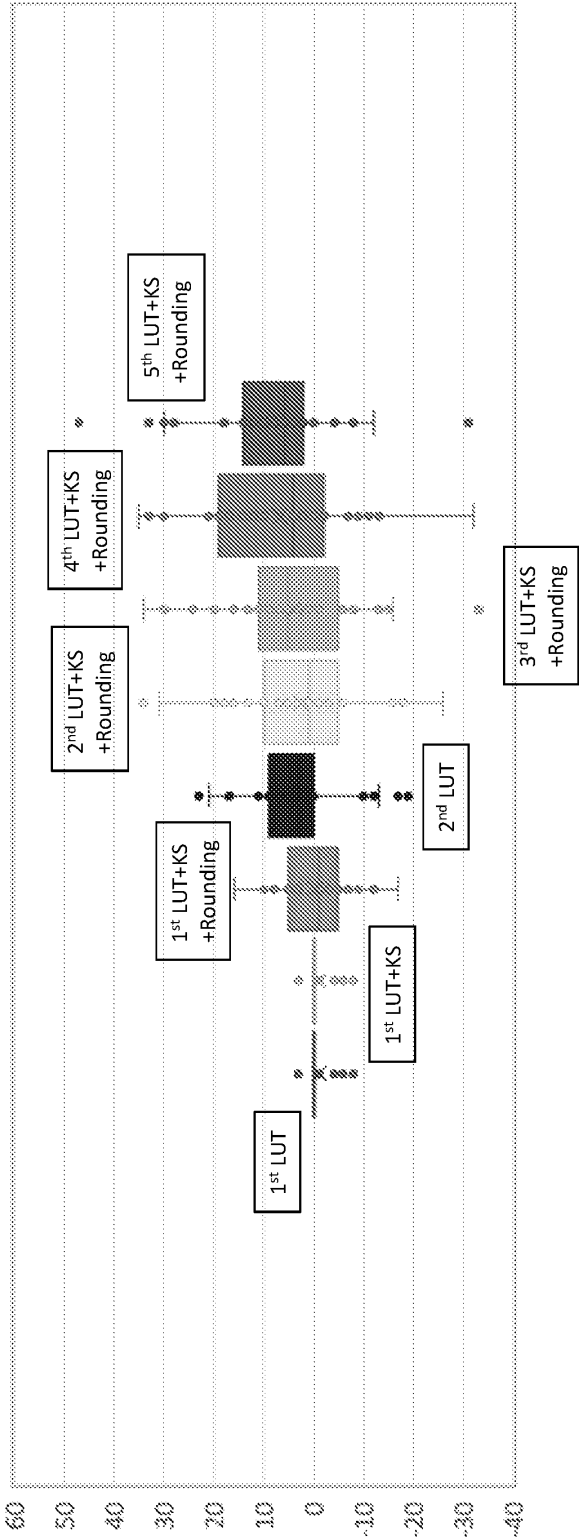


FIG. 34

Table 4: A comparison of our works and other existing works.

Scheme	Summary	Better LUT algorithm	RNS Polynomial multiplications	Specified modulo functions
FHE-DiNN	FHE scheme designed for sign-function deep neural network.	✓	×	-- (Not on integer ring)
PEGASUS	FHE scheme designed for general usage of evaluating functions.	×	✓	×
Present FHE Scheme	FHE scheme designed for many kinds of deep neural network.	✓	×	×
Improved FHE Scheme	FHE scheme designed for many kinds of deep neural network, and for evaluating functions.	✓	✓	✓

FIG. 35

Algorithm 16 NTT multiplication

- 1: **Input:** Polynomials $a, b \in \mathbb{Z}_q[x]/(x^n + 1)$.
- 2: Find $\tilde{a} = NTT(a), \tilde{b} = NTT(b)$. (One $NTT(\cdot)$ costs $O(n \log n)$.)
- 3: Calculate $\tilde{c} = \tilde{a} \times \tilde{b}$. (Position-wise multiplication (PMUL), $O(n)$.)
- 4: Find $c = INTT(\tilde{c})$. (One $INTT(\cdot)$ costs $O(n \log n)$.)
- 5: **Output:** Polynomial c .

FIG. 36

Algorithm 17 Extended RLWE Ciphertext Multiplication ($\diamond$)

Input: plaintext operand r , modulus q , dimension n , gadget decomposition vector $\vec{g} \in \mathbb{Z}^L$, extended RLWE ciphertext $\{(a_i, b_i)\}_{i \in \{0, 1, \dots, L-1\}}$.

Output: RLWE ciphertext $(a, b) \in \text{RLWE}_s^{n,q}(r \times m)$.

Decompose r s.t. $r = \sum_{i=0}^{L-1} r_i \times g_i$.

Compute $(a, b) = \sum_{i=0}^{L-1} r_i \times (a_i, b_i) = \sum_{i=0}^{L-1} (r_i a_i, r_i b_i)$.

Return (a, b) .

FIG. 37

Algorithm 18 RLWE and RGSW Multiplication ($\odot$)

Input: $(a, b) \in \text{RLWE}_s^{n,q}(m_1), (\beta, \alpha) \in \text{RGSW}_s^{n,q}(m_2)$.

Output: $(a', b') \in \text{RLWE}_s^{n,q}(m_1 \times m_2)$.

Return $(a', b') = a \odot \alpha + b \odot \beta$.

FIG. 38

Algorithm 19 CRT based LUT for large modulus.

-
- 1: First turn AC_0 , all RGSW ciphertexts into RNS form. Use overline to represent RNS form: $f(x) \in R_{n',Q} \rightarrow \bar{f} \in \Pi_{i=0}^{L-1} \mathbb{Z}_{q_i}^{n'}$, which includes L vectors.
 - 2: **for** $i = 0, 1, \dots, n/2 - 1$ **do**
 - 3: Calculate $\overline{EK}_i$,
 - 4: $\overline{AC}_{i+1} = \overline{EK}_i \odot \overline{AC}_i + \overline{AC}_i$
 - 5: **end for**
 - 6: **Output:** Extract0($INTT(\overline{AC}_{n/2})$), i.e., the LWE ciphertext of the constant term in the plaintext of $INTT(\overline{AC}_{n/2})$.
-

FIG. 39

Algorithm 20 ($\odot$) operator in RNS form

-
- 1: **Input:** Extended RLWE ciphertext $\bar{\alpha}$ which includes L RLWE ciphertexts, i.e. $2L$ polynomials (in RNS form). According to CRT, find gadget vector $\bar{g} = (g_0, g_1, \dots, g_{L-1}) \in \mathbb{Z}^L$ based on $\{q_i\}$. Let $\bar{\alpha} = \Pi_{i=0}^{L-1} (\bar{a}_i, \bar{b}_i)$, where each $(a_i, b_i) \in \text{RLWE}_{s',Q}^{n',Q}(g_i m)$ for some plaintext m . A RNS form polynomial $\bar{f} \in \Pi_{i=0}^{L-1} \mathbb{Z}_{q_i}^{n'}$.
 - 2: Decompose $\bar{f}$ into $\bar{f}_i = \bar{f} \bmod q_i, i = 0, 1, \dots, L - 1$.
 - 3: Calculate and output $\sum_{i=0}^{L-1} \bar{f}_i(\bar{a}_i, \bar{b}_i)$.
-

FIG. 40

Table 5: Analysis on LUT algorithm

Operations	LUT in the present LUT			Improved LUT			Time saved
	Number of calculation	Time per calculation	Time in total	Number of calculation	Time per calculation	Time in total	
NTT	4354	15.29us	66572us	1792	15.29us	27399 us	39173us
INTT	2177	16.05us	34941us	512	16.05us	8192us	26749us
PMUL	2177	2.84us	6183us	8193	2.84us	23268	-17085us
Module Calculation (After additions)	10625	5.9us	62687us	4094	4.5us	22473us	40214us
Polynomial addition	8192	1.48us	12124us	6144	1.48us	9093us	3031us
Bu-decomposition	512	9.5us	4864us	512	9.5us	4864us	0
Quick multiplication	6144	8us	49152us	0	-	0	49152us
Total time	236523us=237ms			95289us=95ms			142ms (60%)

FIG. 41

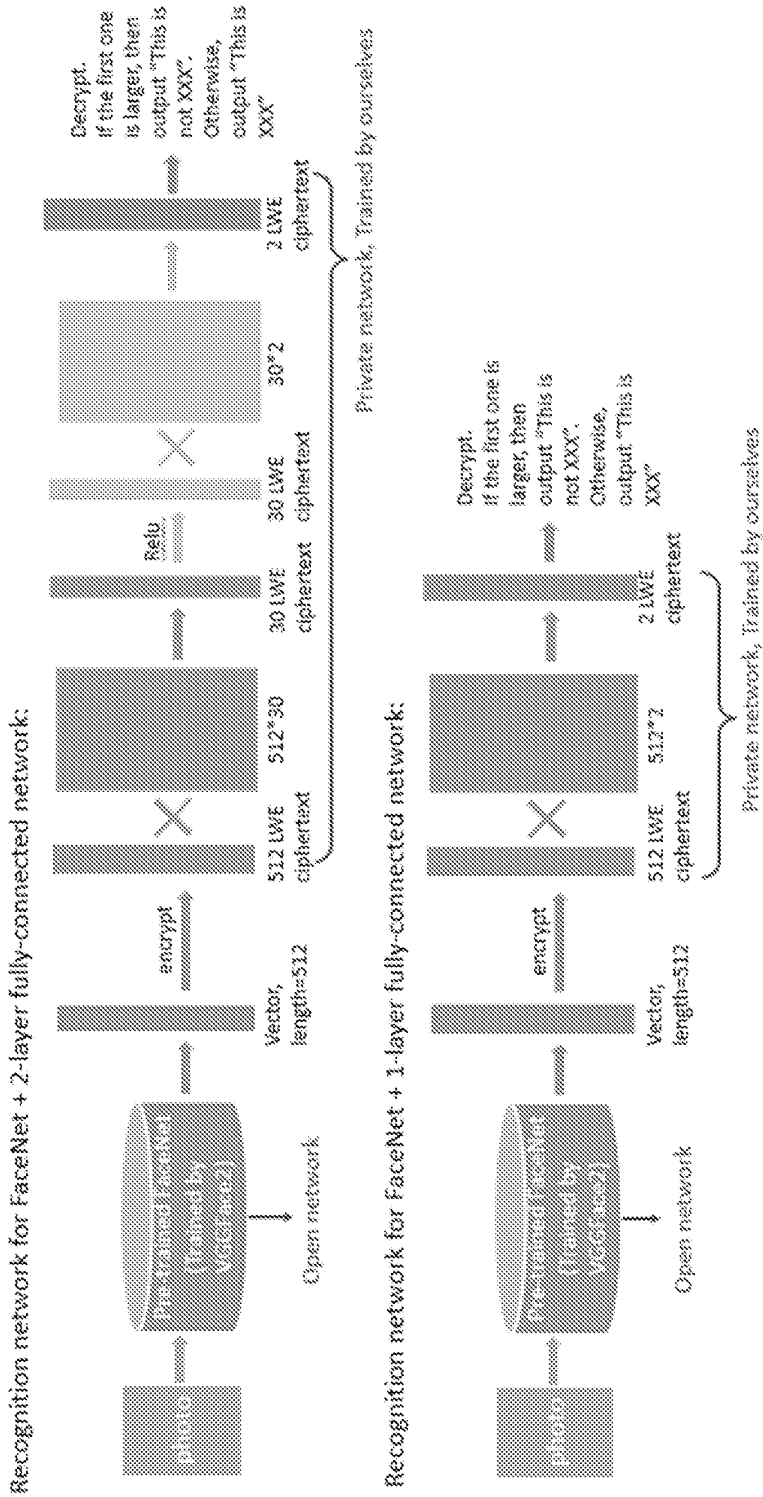


FIG. 42

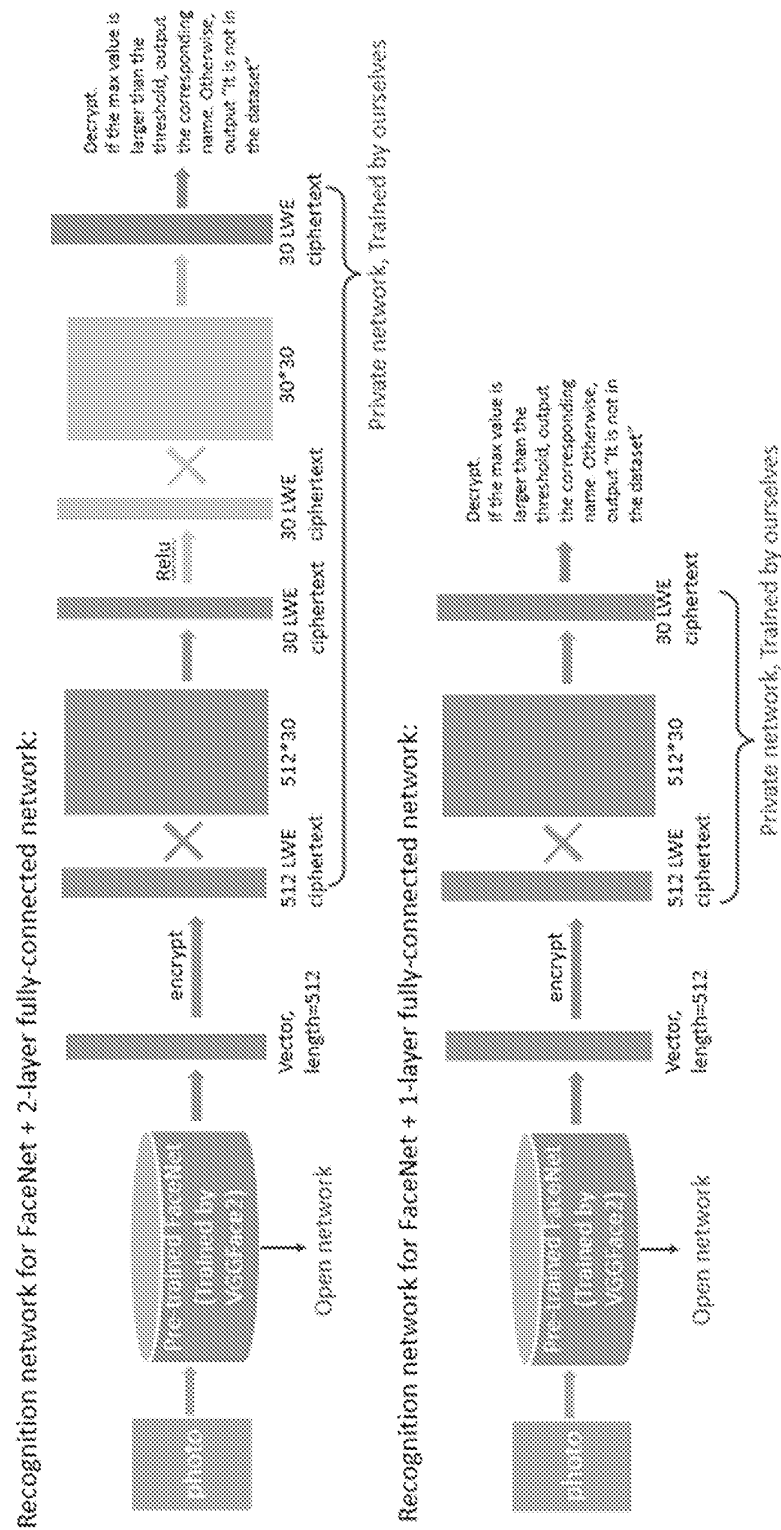


FIG. 43

Table 6: Evaluation results in face recognition

Task	Private network structure	Number of activation function in private network	Time per recognition	Recognition accuracy
1 to 1 face recognition	1-layer fully-connected network	0	0.009s	100%
	2-layer fully-connected network	30	Improved system 0.72s Present system 2.1s	
1 to 30 face recognition	1-layer fully-connected network	0	0.08s	
	2-layer fully-connected network	30	Improved system 0.73s Present system 2.1s	
1 to 100 face recognition	1-layer fully-connected network	0	0.2s	-
	2-layer fully-connected network	100	2.2s	

FIG. 44

Table 7: Evaluation results in BP network (1 hidden layer, 30 nodes)

BP Network (Activation function)	Number of nodes	Prediction accuracy in the plaintext	Homomorphic evaluation system	Time in lab PC	Prediction accuracy
Our model (ReLU)	30	94.80%	Present FHE scheme	2.4s per test (4 threads)	94.04%
Our model (ReLU)	30	94.80%	Improved FHE scheme	0.89s per test (4 threads)	94.04%
Our model (ReLU)	30	94.80%	PEGASUS	100 min for 256 tests (4 threads) About 23s per test	91.39%
FHE-DiNN (sign function)	30	94.76%	FHE-DiNN	10.95s per test (1 thread)	93.71%

FIG. 45