(19)日本国特許庁(JP)

(12)特許公報(B1)

(11)特許番号
特許第7493697号
(P7493697)

(45)発行日　令和6年5月31日(2024.5.31)

(24)登録日　令和6年5月23日(2024.5.23)

(51)国際特許分類		FI	
G06N	*3/0464(2023.01)*	G06N	3/0464
G06N	*3/02 (2006.01)*	G06N	3/02

請求項の数　8　（全43頁）

(21)出願番号　特願2024-519956(P2024-519956)
(86)(22)出願日　令和4年7月1日(2022.7.1)
(86)国際出願番号　PCT/JP2022/026498
審査請求日　令和6年4月1日(2024.4.1)
早期審査対象出願

(73)特許権者　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目７番３号
(74)代理人　110002491
弁理士法人クロスボーダー特許事務所
(72)発明者　西田 悠太郎
東京都千代田区丸の内二丁目７番３号
三菱電機株式会社内
(72)発明者　安田 聖
東京都千代田区丸の内二丁目７番３号
三菱電機株式会社内
(72)発明者　小関 義博
東京都千代田区丸の内二丁目７番３号
三菱電機株式会社内
(72)発明者　花岡 悟一郎
最終頁に続く

(54)【発明の名称】　推論検証システムおよび推論検証方法

(57)【特許請求の範囲】
【請求項１】
推論処理の対象となるデータである小数値を整数値で表現し、前記整数値を畳み込みニューラルネットワークのパラメータとして扱って推論モデルを実行して、前記畳み込みニューラルネットワークの各層の計算結果を含む推論結果を得る推論部と、
前記畳み込みニューラルネットワークの層ごとに、前記層の前記計算結果を入力にして前記層の種類に応じたプロトコルの証明生成アルゴリズムを実行して、前記畳み込みニューラルネットワークの各層に対する前記証明生成アルゴリズムの実行結果を含む証明を得る証明部と、
前記畳み込みニューラルネットワークの層ごとに、前記層の前記実行結果を入力にして前記層の種類に応じた前記プロトコルの検証アルゴリズムを実行して、前記畳み込みニューラルネットワークの各層に対する前記検証アルゴリズムの実行結果を含む検証結果を得る検証部と、
を備える推論検証システム。
【請求項２】
Ｔｈｅ　ＲｅＬＵ　Ｌａｙｅｒプロトコルが、前記畳み込みニューラルネットワークのＲｅＬＵ活性化層に応じた前記プロトコルである
請求項１に記載の推論検証システム。
【請求項３】
Ｔｈｅ　Ａｆｆｉｎｅ　Ｌａｙｅｒプロトコルが、前記畳み込みニューラルネットワーク

のＡｆｆｉｎｅ層に応じた前記プロトコルである
請求項１または請求項２に記載の推論検証システム。
【請求項４】
Ｔｈｅ Ｃｏｎｖｏｌｕｔｉｏｎ Ｌａｙｅｒプロトコルが、前記畳み込みニューラルネットワークの畳み込み層に応じた前記プロトコルである
請求項１または請求項２に記載の推論検証システム。
【請求項５】
Ｔｈｅ Ａｖｅｒａｇｅ Ｐｏｏｌｉｎｇ Ｌａｙｅｒプロトコルが、前記畳み込みニューラルネットワークのＡｖｅｒａｇｅ Ｐｏｏｌｉｎｇ層に応じた前記プロトコルである
請求項１または請求項２に記載の推論検証システム。
【請求項６】
Ｔｈｅ Ｍａｘ Ｐｏｏｌｉｎｇ Ｌａｙｅｒプロトコルが、前記畳み込みニューラルネットワークのＭａｘ Ｐｏｏｌｉｎｇ層に応じた前記プロトコルである
請求項１または請求項２に記載の推論検証システム。
【請求項７】
Ｔｈｅ ＳｏｆｔＭａｘ Ｌａｙｅｒプロトコルが、前記畳み込みニューラルネットワークのＳｏｆｔＭａｘ層に応じた前記プロトコルである
請求項１または請求項２に記載の推論検証システム。
【請求項８】
推論処理の対象となるデータである小数値を整数値で表現し、前記整数値を畳み込みニューラルネットワークのパラメータとして扱って推論モデルを実行して、前記畳み込みニューラルネットワークの各層の計算結果を含む推論結果を得て、
前記畳み込みニューラルネットワークの層ごとに、前記層の前記計算結果を入力にして前記層の種類に応じたプロトコルの証明生成アルゴリズムを実行して、前記畳み込みニューラルネットワークの各層に対する前記証明生成アルゴリズムの実行結果を含む証明を得て、
前記畳み込みニューラルネットワークの層ごとに、前記層の前記実行結果を入力にして前記層の種類に応じた前記プロトコルの検証アルゴリズムを実行して、前記畳み込みニューラルネットワークの各層に対する前記検証アルゴリズムの実行結果を含む検証結果を得る
推論検証方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本開示は、ゼロ知識証明による推論モデルの検証に関するものである。
【背景技術】
【０００２】
ニューラルネットワークによるＡＩ推論技術は、データ分類などの機械学習タスクで大きな成功を収めている。ＡＩは人工知能を略称である。
ニューラルネットワークによるデータ分析を行うためには、事前に大量の学習データを用いて、推論モデルの学習を行う必要がある。この時、学習データを用意することの困難さ及び計算資源の制約などの理由から、ユーザが自身の環境で推論モデルを構築することが困難である場合がある。
こういった観点から、近年ではクラウド上でニューラルネットワークによるデータ分析を提供するサービス（ＭＬａａＳ）がある。このサービスにより、クライアントは、分析したいデータをクラウド上にアップロードすることで、提供されている推論モデルによる推論を実行できる。そのため、クライアントは、推論モデルの構築にコストを費やす必要がない。
ＭＬａａＳは、Ｍａｃｈｉｎｅ Ｌｅａｒｎｉｎｇ ａｓ ａ Ｓｅｒｖｉｃｅの略称である。
【０００３】
ＭＬａａＳにおいて、クライアントが分析したいデータを推論モデルの提供者に送信し

て推論処理を委託する場合、サービス提供者は、クライアントに対して、推論結果が実際に推論モデルによって分析が行われた結果であること、を証明する必要がある。
　最も簡単な解決策は、サービス提供者が推論モデル自体を公開することである。しかし、推論モデルはサービス提供者の知的財産であるため、推論モデルをクライアントに開示することは難しい。
【０００４】
　非特許文献１は、推論モデルによる推論処理が実際に実行されたことを証明することをゼロ知識証明を用いて可能にする方法を提案している。
　この方法により、サービス提供者は、クライアントに対して、推論結果が推論モデルによる分析処理によって得られたものであることを証明することが可能になる。
【先行技術文献】
【非特許文献】
【０００５】
【文献】ｚｋＣＮＮ： Ｚｅｒｏ Ｋｎｏｗｌｅｄｇｅ Ｐｒｏｏｆｓ ｆｏｒ Ｃｏｎｖｏｌｕｔｉｏｎａｌ Ｎｅｕｒａｌ Ｎｅｔｗｏｒｋ Ｐｒｅｄｉｃｔｉｏｎｓ ａｎｄ Ａｃｃｕｒａｃｙ． Ｔｉａｎｙｉ Ｌｉｕ， Ｘｉａｎｇ Ｘｉｅ， ａｎｄ Ｙｕｐｅｎｇ Ｚｈａｎｇ． ２０２１．
【発明の概要】
【発明が解決しようとする課題】
【０００６】
　非特許文献１の方法は、使用されるゼロ知識証明のプロトコルの制約により、推論モデルのパラメータとして整数値しか扱うことができない。
【０００７】
　本開示は、パラメータが小数である推論モデルの検証を可能にすることを目的とする。
【課題を解決するための手段】
【０００８】
　本開示の推論検証システムは、
　推論処理の対象となるデータである小数値を整数値で表現し、前記整数値を畳み込みニューラルネットワークのパラメータとして扱って推論モデルを実行して推論結果を得る推論部と、
　前記推論結果を入力にして証明生成アルゴリズムを実行して証明を得る証明部と、
　前記証明を入力にして検証アルゴリズムを実行して検証結果を得る検証部と、
を備える。
【発明の効果】
【０００９】
　本開示によれば、パラメータが小数である推論モデルの検証が可能になる。
【図面の簡単な説明】
【００１０】
【図１】実施の形態１における推論検証システム１００の構成図。
【図２】実施の形態１におけるパラメータ生成装置２００の構成図。
【図３】実施の形態１における鍵生成装置３００の構成図。
【図４】実施の形態１における推論装置４００の構成図。
【図５】実施の形態１における証明装置５００の構成図。
【図６】実施の形態１における検証装置６００の構成図。
【図７】実施の形態１における推論検証方法（パラメータ生成）のフローチャート。
【図８】実施の形態１における推論検証方法（鍵生成）のフローチャート。
【図９】実施の形態１における推論検証方法（推論）のフローチャート。
【図１０】実施の形態１における推論検証方法（証明）のフローチャート。
【図１１】実施の形態１における推論検証方法（検証）のフローチャート。
【図１２】実施の形態１におけるパラメータ生成装置２００のハードウェア構成図。

【図１３】実施の形態１における鍵生成装置３００のハードウェア構成図。
【図１４】実施の形態１における推論装置４００のハードウェア構成図。
【図１５】実施の形態１における証明装置５００のハードウェア構成図。
【図１６】実施の形態１における検証装置６００のハードウェア構成図。
【発明を実施するための形態】
【００１１】
実施の形態および図面において、同じ要素または対応する要素には同じ符号を付している。説明した要素と同じ符号が付された要素の説明は適宜に省略または簡略化する。図中の矢印はデータの流れ又は処理の流れを主に示している。
【００１２】
実施の形態１．
推論検証システム１００について、図１から図１６に基づいて説明する。
【００１３】
推論検証システム１００は、推論処理を検証することが可能なシステムである。
【００１４】
＊＊＊構成の説明＊＊＊
図１に基づいて、推論検証システム１００の構成を説明する。
推論検証システム１００は、パラメータ生成装置２００と鍵生成装置３００と推論装置４００と証明装置５００と検証装置６００といった装置を備える。
これらの装置は、ネットワークを介して互いに通信する。ネットワークの具体例はインターネットである。
【００１５】
図２に基づいて、パラメータ生成装置２００の構成を説明する。
パラメータ生成装置２００は、プロセッサ２０１とメモリ２０２と補助記憶装置２０３と通信装置２０４と入出力インタフェース２０５といったハードウェアを備えるコンピュータである。これらのハードウェアは、信号線を介して互いに接続されている。
【００１６】
プロセッサ２０１は、パラメータ生成装置２００のプロセッサである。
プロセッサは、演算処理を行うＩＣであり、他のハードウェアを制御する。例えば、プロセッサはＣＰＵである。
ＩＣは、Ｉｎｔｅｇｒａｔｅｄ　Ｃｉｒｃｕｉｔの略称である。
ＣＰＵは、Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔの略称である。
【００１７】
メモリ２０２は、パラメータ生成装置２００のメモリである。
メモリは、揮発性または不揮発性の記憶装置である。メモリは、主記憶装置またはメインメモリとも呼ばれる。例えば、メモリはＲＡＭである。
メモリ２０２に記憶されたデータは必要に応じて補助記憶装置２０３に保存される。
ＲＡＭは、Ｒａｎｄｏｍ　Ａｃｃｅｓｓ　Ｍｅｍｏｒｙの略称である。
【００１８】
補助記憶装置２０３は、パラメータ生成装置２００の補助記憶装置である。
補助記憶装置は不揮発性の記憶装置である。例えば、補助記憶装置は、ＲＯＭ、ＨＤＤ、フラッシュメモリまたはこれらの組み合わせである。
補助記憶装置２０３に記憶されたデータは必要に応じてメモリ２０２にロードされる。
ＲＯＭは、Ｒｅａｄ　Ｏｎｌｙ　Ｍｅｍｏｒｙの略称である。
ＨＤＤは、Ｈａｒｄ　Ｄｉｓｋ　Ｄｒｉｖｅの略称である。
【００１９】
通信装置２０４は、パラメータ生成装置２００の通信装置である。
通信装置はレシーバ及びトランスミッタである。例えば、通信装置は通信チップまたはＮＩＣである。
パラメータ生成装置２００の通信は通信装置２０４を用いて行われる。

ＮＩＣは、Ｎｅｔｗｏｒｋ　Ｉｎｔｅｒｆａｃｅ　Ｃａｒｄの略称である。

【００２０】

入出力インタフェース２０５は、パラメータ生成装置２００の入出力インタフェースである。

入出力インタフェースは、入力装置および出力装置が接続されるポートである。例えば、入出力インタフェースはＵＳＢ端子であり、入力装置はキーボードおよびマウスであり、出力装置はディスプレイである。

パラメータ生成装置２００の入出力は入出力インタフェース２０５を用いて行われる。

ＵＳＢは、Ｕｎｉｖｅｒｓａｌ　Ｓｅｒｉａｌ　Ｂｕｓの略称である。

【００２１】

パラメータ生成装置２００は、受付部２１０と生成部２２０と出力部２３０といった要素を備える。これらの要素はソフトウェアで実現される。

【００２２】

補助記憶装置２０３には、受付部２１０と生成部２２０と出力部２３０としてコンピュータを機能させるためのパラメータ生成プログラムが記憶されている。パラメータ生成プログラムは、メモリ２０２にロードされて、プロセッサ２０１によって実行される。

補助記憶装置２０３には、さらに、ＯＳが記憶されている。ＯＳの少なくとも一部は、メモリにロードされて、プロセッサによって実行される。

プロセッサ２０１は、ＯＳを実行しながら、パラメータ生成プログラムを実行する。

ＯＳは、Ｏｐｅｒａｔｉｎｇ　Ｓｙｓｔｅｍの略称である。

【００２３】

パラメータ生成プログラムの入出力データは記憶部２９０に記憶される。

メモリ２０２は記憶部２９０として機能する。但し、補助記憶装置２０３、プロセッサ２０１内のレジスタおよびプロセッサ２０１内のキャッシュメモリなどの記憶装置が、メモリ２０２の代わりに、又は、メモリ２０２と共に、記憶部２９０として機能してもよい。

【００２４】

パラメータ生成装置２００は、プロセッサ２０１を代替する複数のプロセッサを備えてもよい。

【００２５】

図３に基づいて、鍵生成装置３００の構成を説明する。

鍵生成装置３００は、プロセッサ３０１とメモリ３０２と補助記憶装置３０３と通信装置３０４と入出力インタフェース３０５といったハードウェアを備えるコンピュータである。これらのハードウェアは、信号線を介して互いに接続されている。

【００２６】

プロセッサ３０１は、鍵生成装置３００のプロセッサである。

メモリ３０２は、鍵生成装置３００のメモリである。

補助記憶装置３０３は、鍵生成装置３００の補助記憶装置である。

通信装置３０４は、鍵生成装置３００の通信装置である。

入出力インタフェース３０５は、鍵生成装置３００の入出力インタフェースである。

【００２７】

鍵生成装置３００は、受付部３１０と生成部３２０と出力部３３０といった要素を備える。これらの要素はソフトウェアで実現される。

【００２８】

補助記憶装置３０３には、受付部３１０と生成部３２０と出力部３３０としてコンピュータを機能させるための鍵生成プログラムが記憶されている。鍵生成プログラムは、メモリ３０２にロードされて、プロセッサ３０１によって実行される。

鍵生成装置３００には、さらに、ＯＳが記憶されている。

プロセッサ３０１は、ＯＳを実行しながら、鍵生成プログラムを実行する。

【００２９】

鍵生成プログラムの入出力データは記憶部３９０に記憶される。

メモリ３０２は記憶部３９０として機能する。但し、補助記憶装置３０３、プロセッサ３０１内のレジスタおよびプロセッサ３０１内のキャッシュメモリなどの記憶装置が、メモリ３０２の代わりに、又は、メモリ３０２と共に、記憶部３９０として機能してもよい。

【００３０】

鍵生成装置３００は、プロセッサ３０１を代替する複数のプロセッサを備えてもよい。

【００３１】

図４に基づいて、推論装置４００の構成を説明する。

推論装置４００は、プロセッサ４０１とメモリ４０２と補助記憶装置４０３と通信装置４０４と入出力インタフェース４０５といったハードウェアを備えるコンピュータである。これらのハードウェアは、信号線を介して互いに接続されている。

【００３２】

プロセッサ４０１は、推論装置４００のプロセッサである。

メモリ４０２は、推論装置４００のメモリである。

補助記憶装置４０３は、推論装置４００の補助記憶装置である。

通信装置４０４は、推論装置４００の通信装置である。

入出力インタフェース４０５は、推論装置４００の入出力インタフェースである。

【００３３】

推論装置４００は、受付部４１０と推論部４２０と出力部４３０といった要素を備える。これらの要素はソフトウェアで実現される。

【００３４】

補助記憶装置４０３には、受付部４１０と推論部４２０と出力部４３０としてコンピュータを機能させるための推論プログラムが記憶されている。推論プログラムは、メモリ４０２にロードされて、プロセッサ４０１によって実行される。

推論装置４００には、さらに、ＯＳが記憶されている。

プロセッサ４０１は、ＯＳを実行しながら、推論プログラムを実行する。

【００３５】

推論プログラムの入出力データは記憶部４９０に記憶される。

メモリ４０２は記憶部４９０として機能する。但し、補助記憶装置４０３、プロセッサ４０１内のレジスタおよびプロセッサ４０１内のキャッシュメモリなどの記憶装置が、メモリ４０２の代わりに、又は、メモリ４０２と共に、記憶部４９０として機能してもよい。

【００３６】

推論装置４００は、プロセッサ４０１を代替する複数のプロセッサを備えてもよい。

【００３７】

図５に基づいて、証明装置５００の構成を説明する。

証明装置５００は、プロセッサ５０１とメモリ５０２と補助記憶装置５０３と通信装置５０４と入出力インタフェース５０５といったハードウェアを備えるコンピュータである。これらのハードウェアは、信号線を介して互いに接続されている。

【００３８】

プロセッサ５０１は、証明装置５００のプロセッサである。

メモリ５０２は、証明装置５００のメモリである。

補助記憶装置５０３は、証明装置５００の補助記憶装置である。

通信装置５０４は、証明装置５００の通信装置である。

入出力インタフェース５０５は、証明装置５００の入出力インタフェースである。

【００３９】

証明装置５００は、受付部５１０と保管部５２０と証明部５３０と出力部５４０といった要素を備える。保管部５２０は、鍵保管部５２１と推論結果保管部５２２を備える。これらの要素はソフトウェアで実現される。

【００４０】

補助記憶装置５０３には、受付部５１０と保管部５２０と証明部５３０と出力部５４０としてコンピュータを機能させるための証明プログラムが記憶されている。証明プログラ

ムは、メモリ５０２にロードされて、プロセッサ５０１によって実行される。
証明装置５００には、さらに、ＯＳが記憶されている。
プロセッサ５０１は、ＯＳを実行しながら、証明プログラムを実行する。
【００４１】
証明プログラムの入出力データは記憶部５９０に記憶される。
メモリ５０２は記憶部５９０として機能する。但し、補助記憶装置５０３、プロセッサ５０１内のレジスタおよびプロセッサ５０１内のキャッシュメモリなどの記憶装置が、メモリ５０２の代わりに、又は、メモリ５０２と共に、記憶部５９０として機能してもよい。
【００４２】
証明装置５００は、プロセッサ５０１を代替する複数のプロセッサを備えてもよい。
【００４３】
図６に基づいて、検証装置６００の構成を説明する。
検証装置６００は、プロセッサ６０１とメモリ６０２と補助記憶装置６０３と通信装置６０４と入出力インタフェース６０５といったハードウェアを備えるコンピュータである。これらのハードウェアは、信号線を介して互いに接続されている。
【００４４】
プロセッサ６０１は、検証装置６００のプロセッサである。
メモリ６０２は、検証装置６００のメモリである。
補助記憶装置６０３は、検証装置６００の補助記憶装置である。
通信装置６０４は、検証装置６００の通信装置である。
入出力インタフェース６０５は、検証装置６００の入出力インタフェースである。
【００４５】
検証装置６００は、受付部６１０と保管部６２０と検証部６３０と出力部６４０といった要素を備える。保管部６２０は、鍵保管部６２１と証明保管部６２２を備える。これらの要素はソフトウェアで実現される。
【００４６】
補助記憶装置６０３には、受付部６１０と保管部６２０と検証部６３０と出力部６４０としてコンピュータを機能させるための検証プログラムが記憶されている。検証プログラムは、メモリ６０２にロードされて、プロセッサ６０１によって実行される。
検証装置６００には、さらに、ＯＳが記憶されている。
プロセッサ６０１は、ＯＳを実行しながら、検証プログラムを実行する。
【００４７】
検証プログラムの入出力データは記憶部６９０に記憶される。
メモリ６０２は記憶部６９０として機能する。但し、補助記憶装置６０３、プロセッサ６０１内のレジスタおよびプロセッサ６０１内のキャッシュメモリなどの記憶装置が、メモリ６０２の代わりに、又は、メモリ６０２と共に、記憶部６９０として機能してもよい。
【００４８】
検証装置６００は、プロセッサ６０１を代替する複数のプロセッサを備えてもよい。
【００４９】
＊＊＊記法の説明＊＊＊
以降の説明における記法を示す。
「Ａ」が分布であるときに、「ｙ←Ａ」は、ｙをＡからその分布に従ってランダムに選ぶことを指す。
「Ａ」が集合であるときに、「ｙ←Ａ」は、入力ｘに対して、ｙをＡ”から一様に選ぶことを指す。
「ｐ」は、任意の素数である。
「Ｇ」は、位数ｐの群である。
「ｍ」および「ｎ」は、任意の自然数である。
「ｅ」は、自然対数の底である。
「Ｈ」は、ハッシュ関数である。

「Ｄ」は、推論モデルの入力データの集合である。
「ｘ」は、推論処理の対象となる入力データである。
「Ｍ」は、推論モデルである。
「ｎ」が自然数であるとき、［ｎ］は集合｛１，…，ｎ｝である。
「Ｚ」は、整数の集合である。
「Ｒ」は、実数の集合である。

【００５０】

＊＊＊動作の説明＊＊＊

推論検証システム１００の動作の手順は推論検証方法に相当する。また、推論検証システム１００の動作の手順は推論検証プログラムによる処理の手順に相当する。

推論検証プログラムは、パラメータ生成プログラムと鍵生成プログラムと推論プログラムと証明プログラムと検証プログラムを含む。

各パラメータ生成プログラムは、光ディスクまたはフラッシュメモリ等の不揮発性の記録媒体にコンピュータ読み取り可能に記録（格納）することができる。

【００５１】

図７に基づいて、推論検証方法におけるパラメータ生成装置２００の動作を説明する。

ステップＳ２１０において、受付部２１０は、パラメータ生成装置２００に入力されるパラメータ（λ，Ｄ）を受け付ける。

例えば、パラメータ（λ，Ｄ）は、管理者によってパラメータ生成装置２００に入力される。

【００５２】

ステップＳ２２０において、生成部２２０は、パラメータ（λ，Ｄ）を入力にしてセットアップアルゴリズムを実行する。

これにより、公開パラメータｐｐが生成される。

【００５３】

セットアップアルゴリズムは、公開パラメータｐｐを生成するためのアルゴリズムである。

例えば、セットアップアルゴリズム（Ｓｅｔｕｐ）は、以下のように表される。

【００５４】

【数１】

$$\mathrm{Setup}\left(1^{\lambda}, D\right):$$

$$g_i \leftarrow \mathcal{G}$$

$$\mathrm{pp} \coloneqq (g_1, \ldots, g_n)$$

【００５５】

ステップＳ２３０において、出力部２３０は、公開パラメータｐｐを出力する。

具体的には、出力部２３０は、公開パラメータｐｐを鍵生成装置３００へ送信する。

【００５６】

図８に基づいて、推論検証方法における鍵生成装置３００の動作を説明する。

ステップＳ３１０において、受付部３１０は、鍵生成装置３００に入力される公開パラメータｐｐを受け付ける。

具体的には、受付部３１０は、パラメータ生成装置２００から公開パラメータｐｐを受信する。

【００５７】

ステップＳ３２０において、生成部３２０は、公開パラメータｐｐを入力にして鍵生成アルゴリズムを実行する。
これにより、公開鍵ｐｋと秘密鍵ｓｋが生成される。
【００５８】
鍵生成アルゴリズムは、公開鍵ｐｋと秘密鍵ｓｋを生成するためのアルゴリズムである。
例えば、鍵生成アルゴリズムは（Ｋｇ）は、以下のように表される。
【００５９】
【数２】

$$\mathrm{Kg}(\mathrm{pp}):$$

$$h_i \leftarrow \mathcal{G}$$

$$\mathrm{s_j} \leftarrow Z_p$$

$$\mathrm{pk} \coloneqq (g_1, \ldots, g_n, h_1, \ldots, h_m)$$

$$\mathrm{sk} \coloneqq (s_1, \ldots, s_l)$$

【００６０】
ステップＳ３３０において、出力部３３０は、公開鍵ｐｋと秘密鍵ｓｋを出力する。
具体的には、出力部３３０は、公開鍵ｐｋと秘密鍵ｓｋを証明装置５００へ送信する。また、出力部３３０は、公開鍵ｐｋを検証装置６００へ送信する。
【００６１】
図４に基づいて、推論検証方法における推論装置４００の動作を説明する。
ステップＳ４１０において、受付部４１０は、推論装置４００に入力されるパラメータ（Ｍ，ｘ）を受け付ける。
例えば、推論モデルＭは、推論モデルＭの提供者によって推論装置４００に入力される。また、データｘは、クライアントによって推論装置４００へ送信される。
推論モデルＭは、推論処理のためのモデルである。
データｘは、推論処理の対象となるデータである。
【００６２】
ステップＳ４２０において、推論部４２０は、推論モデルＭとデータｘを入力にして推論処理アルゴリズムを実行する。
これにより、推論結果ｃが生成される。
【００６３】
推論処理アルゴリズムは、推論結果ｃを生成するためのアルゴリズムである。
例えば、推論処理アルゴリズム（Ｃｌａｓｓｉｆｙ）は、以下のように表される。
ＣＮＮは、推論モデルＭによって実行される畳み込みニューラルネットワークである。
【００６４】
【数３】

$$\mathrm{Classify}(M, x):$$

$$c \coloneqq \mathrm{CNN}(x)$$

【００６５】
ステップＳ４３０において、出力部４３０は、推論結果ｃを出力する。
具体的には、出力部４３０は、推論結果ｃを証明装置５００へ送信する。
【００６６】
図１０に基づいて、推論検証方法における証明装置５００の動作を説明する。
ステップＳ５１０において、受付部５１０は、公開鍵ｐｋと秘密鍵ｓｋと推論結果ｃを受け付ける。
具体的には、受付部５１０は、鍵生成装置３００から公開鍵ｐｋと秘密鍵ｓｋを受信する。また、受付部５１０は、推論装置４００から推論結果ｃを受信する。
【００６７】
ステップＳ５２０において、鍵保管部５２１は、公開鍵ｐｋと秘密鍵ｓｋを保管する。
また、推論結果保管部５２２は、推論結果ｃを保管する。
例えば、公開鍵ｐｋと秘密鍵ｓｋと推論結果ｃは、補助記憶装置５０３に記憶される。
【００６８】
ステップＳ５３０において、証明部５３０は、公開鍵ｐｋと秘密鍵ｓｋと推論結果ｃを入力にして証明生成アルゴリズムを実行する。
これにより、証明Ｐが生成される。
【００６９】
証明生成アルゴリズムは、証明Ｐを生成するためのアルゴリズムである。
証明生成アルゴリズム（Ｐｒｏｖｅ）の例について、以下に説明する。
【００７０】
推論結果ｃは、次のように表される。c_iは、ＣＮＮのｉ層目の計算結果である。
$c = c_1, \cdots, c_n$
【００７１】
$Prove_i$は、計算結果c_iに対して証明P_iを生成するためのアルゴリズムである。
例えば、証明生成アルゴリズム（Ｐｒｏｖｅ）は、以下のように表される。
【００７２】
【数４】

$$\mathrm{Prove}(\mathrm{pk}, \mathrm{sk}, c):$$

$$P_1 \coloneqq \mathrm{Prove}_1(pk, sk, c_1), \ldots, P_n \coloneqq \mathrm{Prove}_n(pk, sk, c_n)$$

$$P \coloneqq (P_1, \ldots, P_n)$$

【００７３】
$Prove_i$の種類（P_a～P_f）は、以下の通りである。
（P_a）ＣＮＮの第ｉ層がＲｅＬＵ活性化層である場合、$Prove_i$の種類は$Prove_{ReLU}$である。
（P_b）ＣＮＮの第ｉ層がＡｆｆｉｎｅ層である場合、$Prove_i$の種類は$Prove_{Affine}$である。
（P_c）ＣＮＮの第ｉ層が畳み込み層である場合、$Prove_i$の種類は$Prove_{Conv}$である。
（P_d）ＣＮＮの第ｉ層がＡｖｅｒａｇｅ　Ｐｏｏｌｉｎｇ層である場合、$Prove_i$の種類は$Prove_{AP}$である。
（P_e）ＣＮＮの第ｉ層がＭａｘ　Ｐｏｏｌｉｎｇ層である場合、$Prove_i$の種類は$Prove_{MP}$である。

（Ｐｆ）ＣＮＮの第ｉ層がＳｏｆｔ　Ｍａｘ層である場合、Ｐｒｏｖｅｉの種類はＰｒｏｖｅSoftMaxである。

【００７４】

Ｐｒｏｖｅｉの各種類（Ｐａ～Ｐｆ）について後述する。

【００７５】

ステップＳ５４０において、出力部５４０は、証明Ｐを出力する。

具体的には、出力部５４０は、証明Ｐを検証装置６００へ送信する。

【００７６】

図１１に基づいて、推論検証方法における検証装置６００の動作を説明する。

ステップＳ６１０において、受付部６１０は、公開鍵ｐｋと証明Ｐを受け付ける。

具体的には、受付部６１０は、鍵生成装置３００から公開鍵ｐｋを受信する。また、受付部６１０は、証明装置５００から証明Ｐを受信する。

【００７７】

ステップＳ６２０において、鍵保管部６２１は、公開鍵ｐｋを保管する。

また、証明保管部６２２は、証明Ｐを保管する。

例えば、公開鍵ｐｋと証明Ｐは、補助記憶装置６０３に記憶される。

【００７８】

ステップＳ６３０において、検証部６３０は、公開鍵ｐｋと証明Ｐを入力にして検証アルゴリズムを実行する。

これにより、検証結果Ｖが生成される。

【００７９】

検証アルゴリズムは、検証結果Ｖを生成するためのアルゴリズムである。

検証アルゴリズム（Ｖｅｒｉｆｙ）の例について、以下に説明する。

【００８０】

Ｖｅｒｉｆｙｉは、証明Ｐｉを検証するためのアルゴリズムである。

例えば、検証アルゴリズム（Ｖｅｒｉｆｙ）は、以下のように表される。

【００８１】

【数５】

$$\begin{aligned}&\mathrm{Verify}(\mathrm{pk}, P):\\ &\quad V_1 \coloneqq \mathrm{Verify}_1(pk, P_1), \ldots, \mathrm{Verify}_n(pk, P_n)\\ &\quad \mathrm{if}(V_1 \wedge \cdots \wedge V_n = \mathrm{true})\\ &\qquad V \coloneqq c\\ &\quad \mathrm{if}(V_1 \wedge \cdots \wedge V_n = \mathrm{false})\\ &\qquad V \coloneqq \bot\end{aligned}$$

【００８２】

Ｖｅｒｉｆｙｉの種類（Ｖａ～Ｖｆ）は、以下の通りである。

（Ｖａ）ＣＮＮの第ｉ層がＲｅＬＵ活性化層である場合、Ｖｅｒｉｆｙｉの種類はＶｅｒｉｆｙReLUである。

（Ｖｂ）ＣＮＮの第ｉ層がＡｆｆｉｎｅ層である場合、Ｖｅｒｉｆｙｉの種類はＶｅｒｉｆｙAffineである。

（V_c）ＣＮＮの第ｉ層が畳み込み層である場合、$Verify_i$の種類は$Verify_{Conv}$である。
（V_d）ＣＮＮの第ｉ層がＡｖｅｒａｇｅ　Ｐｏｏｌｉｎｇ層である場合、$Verify_i$の種類は$Verify_{AP}$である。
（V_e）ＣＮＮの第ｉ層がＭａｘ　Ｐｏｏｌｉｎｇ層である場合、$Verify_i$の種類は$Verify_{MP}$である。
（V_f）ＣＮＮの第ｉ層がＳｏｆｔＭａｘ層である場合、$Verify_i$の種類は$Verify_{SoftMax}$である。

【００８３】

$Verify_i$の各種類（V_a～V_f）について後述する。

【００８４】

ステップＳ６４０において、出力部６４０は、検証結果Ｖを出力する。
例えば、出力部６４０は、検証結果Ｖをディスプレイに表示する。

【００８５】

＊＊＊プロトコルおよび整数値表示の説明＊＊＊

実施の形態１で使用されるゼロ知識証明のプロトコルに関して、以下のプロトコルを説明する。また、実施の形態１における小数の整数値表示を説明する。

（１）Ｓｃｈｎｏｒｒプロトコル
（２）複数指数に対するＳｃｈｎｏｒｒプロトコル
（３）一般化Ｓｃｈｎｏｒｒのプロトコル
（４）ＯＲ　Ｐｒｏｏｆプロトコル
（５）ｎＯＲ　Ｐｒｏｏｆプロトコル
（６）小数の整数値表示
（７）Ｒａｎｇｅ　Ｐｒｏｏｆｓプロトコル
（８）Ｍｕｌｔｉｐｌｉｃａｔｉｏｎ　Ｐｒｏｏｆｓプロトコル
（９）Ｔｈｅ　ＲｅＬＵ　Ｌａｙｅｒプロトコル［P_a，V_a］
（１０）Ｔｈｅ　Ａｆｆｉｎｅ　Ｌａｙｅｒプロトコル［P_b，V_b］
（１１）Ｔｈｅ　Ｃｏｎｖｏｌｕｔｉｏｎ　Ｌａｙｅｒプロトコル［P_c，V_c］
（１２）Ｔｈｅ　Ａｖｅｒａｇｅ　Ｐｏｏｌｉｎｇ　Ｌａｙｅｒプロトコル［P_d，V_d］
（１３）Ｔｈｅ　Ｍａｘ　Ｐｏｏｌｉｎｇ　Ｌａｙｅｒプロトコル［P_e，V_e］
（１４）Ｔｈｅ　ＳｏｆｔＭａｘ　Ｌａｙｅｒプロトコル［P_f，V_f］

【００８６】

（１）Ｓｃｈｎｏｒｒプロトコルについて説明する。
「ｇ」がＧの生成元である。このとき、$x \in Z_p$に対して$y = g^x$が成り立つ。
Ｓｃｈｎｏｒｒプロトコルは、検証者に対してｘに関する情報を一切与えずに、証明者が$y = g^x$におけるｘを知っていることを証明するためのプロトコルである。
Ｓｃｈｎｏｒｒプロトコルは、証明生成アルゴリズム$Prove_{Schnorr}$と、検証アルゴリズム$Verify_{Schnorr}$と、で構成される。

【００８７】

$Prove_{Schnorr}$は、証明Ｐを出力する。

【００８８】

【数１１】

$$\mathrm{Prove}_{\mathrm{Schnorr}}(g, y, x):$$

$$r \leftarrow Z_p$$

$$R \coloneqq g^r$$

$$C \coloneqq H(g, y, R)$$

$$s \coloneqq r + Cx$$

$$P = (g, y, C, s)$$

【００８９】
Ｖｅｒｉｆｙ$_{\mathrm{Schnorr}}$は、ｔｒｕｅまたはｆａｌｓｅを出力する。
【００９０】
【数１２】

$$\mathrm{Verify}_{\mathrm{Schnorr}}(g, y, C, s):$$

$$HV \coloneqq H(g, y, g^s / y^C)$$

$$\mathrm{if}(C = HV) \quad \mathrm{true}$$

$$\mathrm{if}(C \neq HV) \quad \mathrm{false}$$

【００９１】
（２）複数指数に対するＳｃｈｎｏｒｒプロトコルについて説明する。複数指数に対するＳｃｈｎｏｒｒプロトコルは、（１）Ｓｈｎｏｒｒプロトコルを一般化して得られる。
ｇ$_1$，・・・，ｇ$_n$がＧの生成元である。このとき、ｘ$_1$∈Ｚ$_p$，・・・，ｘ$_n$∈Ｚ$_p$に対して式（２Ａ）が成り立つ。
【００９２】
【数１３】

$$y = \Pi_{i=1}^{n} g_i^{x_i} \qquad (2\mathrm{A})$$

【００９３】
複数指数に対するＳｃｈｎｏｒｒプロトコルは、検証者に対して（ｘ$_1$，・・・，ｘ$_n$）に関する情報を一切与えずに、証明者が式（２Ａ）における（ｘ$_1$，・・・，ｘ$_n$）を知っていることを証明するためのプロトコルである。
複数指数に対するＳｃｈｎｏｒｒプロトコルは、証明作成アルゴリズムＰｒｏｖｅ$_{\mathrm{MultiSchnorr}}$と、検証アルゴリズムＶｅｒｉｆｙ$_{\mathrm{MultiSchnorr}}$と、で構成される。

【００９４】
Ｐｒｏｖｅ$_{\mathrm{MultiSchnorr}}$は、証明Ｐを出力する。
【００９５】
【数１４】

$$\mathrm{Prove}_{\mathrm{MultiSchnorr}}(g_1,\dots,g_n,y,x_1,\dots,x_n):$$

$$r_i \leftarrow Z_p \quad (i \in \{0,\dots,n\})$$

$$R \coloneqq \prod_{i=1}^{n} g_i^{r_i}$$

$$C \coloneqq H(g_1,\dots,g_n,y,R)$$

$$s_i \coloneqq r_i + Cx_i \quad (i \in \{0,\dots,n\})$$

$$P = (g_1,\dots,g_n,y,C,s_1,\dots,s_n)$$

【００９６】
Ｖｅｒｉｆｙ$_{\mathrm{MultiSchnorr}}$は、ｔｒｕｅまたはｆａｌｓｅを出力する。
【００９７】
【数１５】

$$\mathrm{Verify}_{\mathrm{MultiSchnorr}}(g_1,\dots,g_n,y,C,s_1,\dots,s_n):$$

$$HV \coloneqq H\left(g_1,\dots,g_n,y,\prod_{i=1}^{n} g_i^{s_i}/y^C\right)$$

$$\mathrm{if}(C = HV) \quad \mathrm{true}$$

$$\mathrm{if}(C \neq HV) \quad \mathrm{false}$$

【００９８】
（３）一般化Ｓｃｈｎｏｒｒプロトコルについて説明する。一般化Ｓｃｈｎｏｒｒプロトコルは、（２）複数指数に対するＳｈｎｏｒｒプロトコルを一般化して得られる。
ｇ$_{1,1}$，・・・，ｇ$_{1,n}$，ｇ$_{2,1}$，・・・，ｇ$_{m,n}$がＧのｍｎ個の生成元である。このとき、ｘ$_1$∈Ｚ$_p$，・・・，ｘ$_n$∈Ｚ$_p$に対して式（３Ａ）が成り立つ。
【００９９】
【数１６】

$$\bigwedge_{j=1}^{m}\left(\prod_{i=1}^{n} g_{j,i}^{x_i} = y_j\right) \qquad (3A)$$

【０１００】
一般化Ｓｃｈｎｏｒｒプロトコルは、検証者に対して（ｘ１，・・・，ｘｎ）に関する情報を一切与えずに、証明者が式（３Ａ）における（ｘ１，・・・，ｘｎ）を知っていることを証明するためのプロトコルである。
一般化Ｓｃｈｎｏｒｒプロトコルは、証明作成アルゴリズムＰｒｏｖｅＧｅｎＳｃｈｎｏｒｒと、検証アルゴリズムＶｅｒｉｆｙＧｅｎＳｃｈｎｏｒｒと、で構成される。
【０１０１】
ＰｒｏｖｅＧｅｎＳｃｈｎｏｒｒは、証明Ｐを出力する。
【０１０２】
【数１７】

$$\mathrm{Prove}_{\mathrm{GenSchnorr}}\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y_j\right)_{j\in[m]},\left(x_i\right)_{i\in[n]}\right):$$

$$r_1,\ldots,\mathrm{r_n}\leftarrow Z_p$$

$$R_1 \coloneqq \prod_{i=1}^{n} g_{1,i}^{r_i},\ldots,R_m \coloneqq \prod_{i=1}^{n} g_{m,i}^{r_i}$$

$$C \coloneqq H\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y_j\right)_{j\in[m]},\left(R_j\right)_{j\in[m]}\right)$$

$$s_1 \coloneqq r_1 + Cx_1,\ldots,s_n \coloneqq r_n + Cx_n$$

$$P \coloneqq \left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y_j\right)_{j\in[m]},C,s_1,\ldots,s_n\right)$$

【０１０３】
ＶｅｒｉｆｙＧｅｎＳｃｈｎｏｒｒは、ｔｒｕｅまたはｆａｌｓｅを出力する。
【０１０４】
【数１８】

$$\mathrm{Verify}_{\mathrm{GenSchnorr}}\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y_j\right)_{j\in[m]},C,s_1,\ldots,s_n\right):$$

$$HV \coloneqq H\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y_j\right)_{j\in[m]},\left(\Pi_{i=1}^{n} g_{j,i}^{s_i}/{y_j}^{C}\right)_{j\in[m]}\right)$$

$$\mathrm{if}(C = HV) \quad \mathrm{true}$$

$$\mathrm{if}(C \neq HV) \quad \mathrm{false}$$

【０１０５】
（４）ＯＲ　Ｐｒｏｏｆプロトコルについて説明する。
ｇ１，１，・・・，ｇ１，ｎ，ｇ２，１，・・・，ｇｍ，ｎがＧのｍｎ個の生成元である。このとき、ｘ＝（ｘ１，・・・，ｘｎ）に対して式（４Ａ）が成り立つ。

【０１０６】
【数１９】

$$f(x) = \left(\prod_{i=1}^{n} {g_{j,i}}^{x_i} \right)_{j=1,\ldots,m} \qquad (4A)$$

【０１０７】
また、ｉ＝１，２に対して以下の式が成り立つ。
$x^{(i)} = (x_1^{(i)}, \cdots, x_n^{(i)}) \in (Z_p)^n$
$y^{(i)} = f(x^{(i)})$
【０１０８】
ＯＲ　Ｐｒｏｏｆプロトコルは、検証者に対して$x^{(1)}$と$x^{(2)}$に関する情報を一切与えずに、証明者が$y^{(1)} = f(x^{(1)})$における$x^{(1)}$、または、$y^{(2)} = f(x^{(2)})$における$x^{(2)}$を知っていることを証明するためのプロトコルである。
ＯＲ　Ｐｒｏｏｆのプロトコルは、証明作成アルゴリズムProve_{OR}と、検証アルゴリズムVerify_{OR}と、で構成される。
【０１０９】
Prove_{OR}は、証明Ｐを出力する。
【０１１０】
【数２０】

$$\mathrm{Prove}_{\mathrm{OR}}\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y^{(i)},x^{(i)}\right)_{i=1,2}\right):$$

$$j \coloneqq 3-i \quad (i=1 \text{ or } 2)$$

$$s^{(j)} \leftarrow \left(Z_p\right)^n$$

$$C^{(j)} \leftarrow Z_p$$

$$R^{(j)} \coloneqq f\left(s^{(j)}\right)/\left(y^{(j)}\right)^{c^{(j)}}$$

$$r^{(i)} \leftarrow \left(Z_p\right)^n$$

$$R^{(i)} \coloneqq f\left(r^{(i)}\right)$$

$$C \coloneqq H\left(y^{(1)},y^{(2)},R^{(1)},R^{(2)}\right)$$

$$C^{(i)} \coloneqq C-C^{(j)}$$

$$s^{(i)} \coloneqq r^{(i)}+C^{(i)}x^{(i)}$$

$$P \coloneqq \left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},y^{(1)},y^{(2)},C^{(1)},C^{(2)},s^{(1)},s^{(2)}\right)$$

【０１１１】

Ｖｅｒｉｆｙ$_{\mathrm{OR}}$は、ｔｒｕｅまたはｆａｌｓｅを出力する。

【０１１２】

【数２１】

$$\mathrm{Verify}_{\mathrm{OR}}\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},y^{(1)},y^{(2)},C^{(1)},C^{(2)},s^{(1)},s^{(2)}\right):$$

$$HV \coloneqq H\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]}y^{(1)},y^{(2)},f\left(s^{(1)}\right)/\left(y^{(1)}\right)^{c^{(1)}},f\left(s^{(2)}\right)/\left(y^{(2)}\right)^{c^{(2)}}\right)$$

$$\mathrm{if}\left(C^{(1)}+C^{(2)}=HV\right) \quad \mathrm{true}$$

$$\mathrm{if}\left(C^{(1)}+C^{(2)} \neq HV\right) \quad \mathrm{false}$$

【０１１３】

（５）ｎＯＲ　Ｐｒｏｏｆプロトコルについて説明する。ｎＯＲ　Ｐｒｏｏｆプロトコルは、（４）ＯＲ　Ｐｒｏｏｆプロトコルを一般化して得られる。

$g_{1,1}$，・・・，$g_{1,n}$，$g_{2,1}$，・・・，$g_{m,n}$がＧのｍｎ個の生成元である。このとき、$x=(x_1,\cdots,x_n)$に対して式（５Ａ）が成り立つ。

【０１１４】

【数２２】

$$f(x)=\left(\prod_{i=1}^{n} {g_{j,i}}^{x_i}\right)_{j=1,\ldots,m} \qquad (5A)$$

【０１１５】

また、$i\in[n]$に対して以下の式が成り立つ。

$x^{(i)}=(x_1^{(i)},\cdots,x_n^{(i)})\in(Z_p)^n$

$y^{(i)}=f(x^{(i)})$

【０１１６】

ｎＯＲ　Ｐｒｏｏｆプロトコルは、検証者に対して各$x^{(i)}$に関する情報を一切与えずに、証明者が$y^{(i)}=f(x^{(i)})$における$x^{(i)}$のうちの少なくとも１つは知っていることを証明するためのプロトコルである。

ｎＯＲ　Ｐｒｏｏｆプロトコルは、証明作成アルゴリズムProven_{OR}と、検証アルゴリズム$\mathrm{Verifyn}_{OR}$と、で構成される。

Proven_{OR}は、全ての$j\in[n]\setminus\{i\}$に対してｓ（ｊ）とＣ（ｊ）とＲ（ｊ）の３つの式を計算し、証明Ｐを出力する。

【０１１７】

【数２３】

$$\mathrm{Prove}_{\mathrm{nOR}}\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y^{(i)},x^{(i)}\right)_{i\in[n]}\right):$$

$$s^{(j)} \leftarrow \left(Z_p\right)^n$$

$$C^{(j)} \leftarrow Z_p$$

$$R^{(j)} \coloneqq f\left(s^{(j)}\right)/\left(y^{(j)}\right)^{C^{(j)}}$$

$$r^{(i)} \leftarrow \left(Z_p\right)^n$$

$$R^{(i)} \coloneqq f\left(r^{(i)}\right)$$

$$C \coloneqq H\left(\left(y^{(i)},R^{(i)}\right)_{i\in[n]}\right)$$

$$C^{(i)} \coloneqq C-\sum_{j\in[n]\backslash\{i\}} C^{(j)}$$

$$s^{(i)} \coloneqq r^{(i)}+C^{(i)}x^{(i)}$$

$$P \coloneqq \left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y^{(i)},C^{(i)},s^{(i)}\right)_{i\in[n]}\right)$$

【０１１８】
Ｖｅｒｉｆｙ$_{\mathrm{nOR}}$は、ｔｒｕｅまたはｆａｌｓｅを出力する。
【０１１９】
【数２４】

$$\mathrm{Verify}_{\mathrm{nOR}}\left(\left(g_{j,i}\right)_{j\in[m],i\in[n]},\left(y^{(i)},C^{(i)},s^{(i)}\right)_{i\in[n]}\right):$$

$$HV \coloneqq H\left(\left(y^{(i)},f\left(s^{(i)}\right)/\left(y^{(i)}\right)^{C^{(i)}}\right)_{i\in[n]}\right)$$

$$\mathrm{if}\left(C^{(1)}+\cdots+C^{(n)}=HV\right)\quad \mathrm{true}$$

$$\mathrm{if}\left(C^{(1)}+\cdots+C^{(n)}\neq HV\right)\quad \mathrm{false}$$

【０１２０】
（６）小数の整数値表示について説明する。
「ｌ」，「ｄ」は正の整数であり、式（６Ａ）が成り立つ。

【０１２１】
【数２５】

$$2^{2(l+d)+1} + 2^d \leq (p-1)/2 \qquad (6A)$$

【０１２２】
このとき、小数ｘの固定点表示は整数＜Ｘ＞で表現される。
【０１２３】
【数２６】

$$x = x_0 + x_1 2^{-d} \quad (x_0 \in \{-2^l, \dots, 2^l - 1\}, x_1 \in \{0, \dots, 2^d - 1\})$$

$$\langle x \rangle = x_0 2^d + x_1 \in Z_p$$

【０１２４】
これにより、通常は実数で表現される畳み込みニューラルネットワークのモデルパラメータを、整数値として扱うことが可能になる。
【０１２５】
（７）Ｒａｎｇｅ　Ｐｒｏｏｆｓプロトコルについて説明する。
「ｇ」，「ｈ」は、Ｇの生成元である。
「ｔ」は、整数である。
Ｒａｎｇｅ　Ｐｒｏｏｆｓプロトコルは、整数ｔに関する情報を検証者に与えることなく、整数ｔが０以上２^{m}未満の整数であることを証明者が証明するためのプロトコルである。
Ｒａｎｇｅ　Ｐｒｏｏｆｓプロトコルは、証明作成アルゴリズムＰｒｏｖｅ$_{Range}$と、検証アルゴリズムＶｅｒｉｆｙ$_{Range}$と、で構成される。
【０１２６】
Ｐｒｏｖｅ$_{Range}$は、ｔの２進数表示の式を計算する。
【０１２７】
【数２７】

$$t = \sum_{i=1}^{m-1} 2^i t_i \ (t_i \in 0,1)$$

$$s_1 \leftarrow Z_p, \dots, s_{m-1} \leftarrow Z_p$$

$$B_1 \coloneqq g^{s_1} h^{t_1}, \dots, B_{m-1} \coloneqq g^{s_{m-1}} h^{t_{m-1}}$$

【０１２８】
そして、Ｐｒｏｖｅ$_{Range}$は、（３）一般化ＳｃｈｎｏｒｒプロトコルにおけるプロトコルＰｒｏｖｅ$_{GenSchnorr}$を用いて、以下の４つの式に対する証明Ｐを算出する。
【０１２９】

【数２８】

$$\mathrm{Prove}_{\mathrm{Range}}(g,h,t):$$

$$A = g^{s}h^{t}$$

$$B_i = g^{s_i}h^{t_i} \quad (i \in \{0,\dots,m-1\})$$

$$B_i^{t_i}/B_i = g^{s_i t_i - s_i} \quad (i \in \{0,\dots,m-1\})$$

$$A = g^{s}\prod_{i=1}^{m-1} h^{2^i t_i}$$

【０１３０】

Ｖｅｒｉｆｙ$_{\mathrm{Range}}$は、検証結果Ｖを出力する。

【０１３１】

【数２９】

$$\mathrm{Verify}_{\mathrm{Range}}(P):$$

$$V \coloneqq \mathrm{Verify}_{\mathrm{GenSchnorr}}(P)$$

【０１３２】

（８）Ｍｕｌｔｉｐｌｉｃａｔｉｏｎ Ｐｒｏｏｆｓプロトコルについて説明する。

「ｇ」，「ｈ」は、Ｇの生成元である。このとき、整数ｘ，ｙ，ｚに対して以下の式が成り立つ。

＜ｘ＞＝ｘ$_0$＋ｘ$_1$$^{2^d}$

＜ｙ＞＝ｙ$_0$＋ｙ$_1$$^{2^d}$

＜ｚ＞＝ｚ$_0$＋ｚ$_1$$^{2^d}$

Ｍｕｌｔｉｐｌｉｃａｔｉｏｎ Ｐｒｏｏｆｓプロトコルは、ｚ、ｘ、ｙに関する情報を検証者に与えることなく、整数ｘ、ｙ、ｚに対して＜ｚ＞＝＜ｘｙ＞の関係が切り捨てを除いて成立することを証明者が証明するためのプロトコルである。

Ｍｕｌｔｉｐｌｉｃａｔｉｏｎ Ｐｒｏｏｆｓプロトコルは、証明作成アルゴリズムＰｒｏｖｅ$_{\mathrm{Mult}}$と、検証アルゴリズムＶｅｒｉｆｙ$_{\mathrm{Mult}}$と、で構成される。

【０１３３】

Ｐｒｏｖｅ$_{\mathrm{Mult}}$において、各符号の意味は以下の通りである。

【０１３４】

【数３０】

$$r_x, r_y, r_z \leftarrow Z_p$$

$$w \coloneqq x_1 y_1 \bmod 2^d$$

$$r_w \leftarrow Z_p$$

$$C_w \coloneqq g^{r_w} h^w$$

【０１３５】
ＰｒｏｖｅＭｕｌｔは、（３）一般化Ｓｃｈｎｏｒｒプロトコルと（７）Ｒａｎｇｅ Ｐｒｏｏｆｓプロトコルを用いて、以下の９つの式に対する証明Ｐを算出する。
【０１３６】
【数３１】

$$\mathrm{Prove}_{\mathrm{Mult}}(g, h, x, y, z):$$

$$C_x = g^{r_x} h^{\langle x \rangle}$$

$$C_y = g^{r_y} h^{\langle y \rangle}$$

$$\langle x \rangle \in \{-2^{l+d}, \ldots, 2^{l+d} - 1\}$$

$$\langle y \rangle \in \{-2^{l+d}, \ldots, 2^{l+d} - 1\}$$

$$C_z = g^{r_z} h^{\langle z \rangle}$$

$$\langle z \rangle \in \{-2^{l+d}, \ldots, 2^{l+d} - 1\}$$

$$C_w = g^{r_w} h^{\langle w \rangle}$$

$$\langle w \rangle \in \{-2^{l+d}, \ldots, 2^{l+d} - 1\}$$

$$C_x^{\langle y \rangle} / \left(C_z^{2^d} h^w \right) = g^{r_x \langle y \rangle - r_z 2^d}$$

【０１３７】
ＶｅｒｉｆｙＭｕｌｔは、検証結果Ｖを出力する。
【０１３８】
【数３２】

$$\mathrm{Verify}_{\mathrm{Mult}}(P):$$

$$V \coloneqq \mathrm{Verify}_{\mathrm{GenSchnorr}}(P)$$

【０１３９】
（９）Ｔｈｅ　ＲｅＬＵ　Ｒａｙｅｒプロトコルについて説明する。
ＲｅＬＵ関数は、畳み込みニューラルネットワークの活性化層で用いられる関数である。
【０１４０】
【数３３】

$$\mathrm{ReLU}(x) = \begin{cases} 0 & (x < 0) \\ x & (0 \leq x) \end{cases}$$

【０１４１】
Ｔｈｅ　ＲｅＬＵ　Ｒａｙｅｒプロトコルは、整数ｘ、ｙに関する情報を検証者に与えることなく、整数ｘ、ｙに対してｙ＝ＲｅＬＵ（ｘ）の関係が成立することを証明者が証明するためのプロトコルである。
【０１４２】
Ｔｈｅ　ＲｅＬＵ　Ｒａｙｅｒプロトコルは、（Ｐａ）証明生成アルゴリズムＰｒｏｖｅ$_{\mathrm{ReLU}}$と、（Ｖａ）検証アルゴリズムＶｅｒｉｆｙ$_{\mathrm{ReLU}}$と、で構成される。
【０１４３】
Ｐｒｏｖｅ$_{\mathrm{ReLU}}$において、各符号の意味は以下の通りである。
【０１４４】
【数３４】

$$a \coloneqq 0$$

$$r_x, r_y, r_a, \mathrm{r_{neg}} \leftarrow Z_p$$

$$C_{neg} \coloneqq g^{r_{neg}} h^{\langle ax \rangle}$$

【０１４５】
Ｐｒｏｖｅ$_{\mathrm{ReLU}}$は、（４）ＯＲ　ｐｒｏｏｆプロトコルと（７）Ｒａｎｇｅ　Ｐｒｏｏｆプロトコルと（８）Ｍｕｌｔｉｐｌｉｃａｔｉｏｎ　Ｐｒｏｏｆｓプロトコルを用いて、以下の式に対する証明Ｐを算出する。
【０１４６】
【数３５】

$$\mathrm{Prove}_{\mathrm{ReLU}}(g,h,x,y):$$

$$C_x = g^{r_x}h^{\langle x\rangle}$$

$$\langle x\rangle \in \{-2^{l+d},\dots,2^{l+d}-1\}$$

$$C_y = g^{r_y}h^{\langle y\rangle}$$

$$\langle y\rangle \in \{-2^{l+d},\dots,2^{l+d}-1\}$$

$$C_a = g^{r_a}h^{\langle a\rangle}$$

$$\langle a\rangle \in \{-2^{l+d},\dots,2^{l+d}-1\}$$

$$\langle ax\rangle = \langle a\rangle\langle x\rangle$$

【０１４７】
証明Ｐは、次の２つの式のどちらかが成立することを証明する。
【０１４８】
【数３６】

$$\langle x\rangle \in \{-2^{l+d},\dots,2^{l+d}-1\} \wedge C_{neg}/C_y = g^{\delta}$$

$$\langle x\rangle \in \{0,\dots,2^{l+d}-1\} \wedge C_x/C_y = g^{\delta}$$

【０１４９】
$\mathrm{Verify}_{\mathrm{ReLU}}$は、検証結果Ｖを出力する。
【０１５０】
【数３７】

$$\mathrm{Verify}_{\mathrm{ReLU}}(P):$$

$$V \coloneqq \mathrm{Verify}_{\mathrm{OR}}(P)$$

【０１５１】
（１０）Ｔｈｅ　Ａｆｆｉｎｅ　Ｌａｙｅｒプロトコルについて説明する。
Ｔｈｅ　Ａｆｆｉｎｅ　Ｌａｙｅｒプロトコルは、ｘ、ｙ、Ａ、ｂに関する情報を検証者に与えることなく、$A \in R^{n\times n}$，$b \in R^{n}$，x，$y \in R^{n}$に対して$y = Ax + b$が成り立

つことを証明者が証明するためのプロトコルである。
Ａ，ｂ，ｘ，ｙは、以下の通りである。
【０１５２】
【数３８】

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n} \end{pmatrix},$$

$$b = (b_1, \ldots, b_n), x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n)$$

【０１５３】
Ｔｈｅ Ａｆｆｉｎｅ Ｌａｙｅｒ Ｐｒｏｔｏｃｏｌは、（Ｐｂ）証明生成アルゴリズムＰｒｏｖｅＡｆｆｉｎｅと、（Ｖｂ）検証アルゴリズムＶｅｒｉｆｙＡｆｆｉｎｅと、で構成される。
【０１５４】
ＰｒｏｖｅＡｆｆｉｎｅにおいて、各符号の意味は以下の通りである。
【０１５５】
【数３９】

$$r_{a_{i,j}}, r_{x_i}, r_{y_i}, \mathrm{r_{i,j}}, r_{b_i} \leftarrow Z_p \quad ((i,j) \in [n] \times [n])$$

$$C'_{i,j} := g^{r_{i,j}} h^{\langle a_i x_j \rangle} \quad ((i,j) \in [n] \times [n])$$

【０１５６】
ＰｒｏｖｅＡｆｆｉｎｅは、（３）一般化Ｓｈｎｏｒｒプロトコルと（７）Ｒａｎｇｅ Ｐｒｏｏｆｓプロトコルと（８）Ｍｕｌｔｉｐｌｉｃａｔｉｏｎ Ｐｒｏｏｆｓプロトコルを用いて、以下の式に対する証明Ｐを算出する。
【０１５７】
【数４０】

$\mathrm{Prove}_{\mathrm{Affine}}(g, h, x, y, b, A)$:

$$C_{a_{i,j}} = g^{r_{a_{i,j}}} h^{\langle a_{i,j} \rangle} \ ((i,j) \in [n] \times [n]),$$

$$\langle a_{i,j} \rangle \in \{-2^{l+d}, \ldots, 2^{l+d} - 1\} \ ((i,j) \in [n] \times [n]),$$

$$C_{b_i} = g^{r_{b_i}} h^{\langle b_i \rangle} \ (i \in [n]),$$

$$\langle b_i \rangle \in \{-2^{l+d}, \ldots, 2^{l+d} - 1\} \ (i \in [n]),$$

$$C_{x_i} = g^{r_{x_i}} h^{\langle x_i \rangle} \ (i \in [n]),$$

$$\langle x_i \rangle \in \{-2^{l+d}, \ldots, 2^{l+d} - 1\} \ (i \in [n]),$$

$$C_{y_i} = g^{r_{y_i}} h^{\langle y_i \rangle} \ (i \in [n]),$$

$$\langle y_i \rangle \in \{-2^{l+d}, \ldots, 2^{l+d} - 1\} \ (i \in [n]),$$

$$\langle a_{i,j} x_i \rangle = \langle a_{i,j} \rangle \langle x_i \rangle \ ((i,j) \in [n] \times [n]),$$

$$C_{y_i} / \left(\prod_{j=1}^{n} C'_{i,j} \right) C_{b_i} = g^{\delta_i} \ (i \in [n]).$$

【０１５８】
Ｖｅｒｉｆｙ$_{\mathrm{Affine}}$は、検証結果Ｖを出力する。
【０１５９】
【数４１】

$$\mathrm{Verify}_{\mathrm{ReLU}}(P):$$

$$V \coloneqq \mathrm{Verify}_{\mathrm{GenShnorr}}(P)$$

【０１６０】
（１１）Ｔｈｅ　Ｃｏｎｖｏｌｕｔｉｏｎ　Ｌａｙｅｒプロトコルについて説明する。
Ｔｈｅ　Ｃｏｎｖｏｌｕｔｉｏｎ　Ｌａｙｅｒプロトコルは、ｘ、ｙ、Ｃｏｎｖに関する情報を検証者に与えることなく、ｘ，ｙ∈Ｒ$^{mm'}$に対してｙ＝Ｃｏｎｖ（ｘ）が成り立つことを証明者が証明するためのプロトコルである。
Ｃｏｎｖは、畳み込みニューラルネットワークの畳み込み層で入力データｘに対して実行される演算である。

Ｔｈｅ Ｃｏｎｖｏｌｕｔｉｏｎ Ｌａｙｅｒプロトコルは、（Ｐ$_{c}$）証明生成アルゴリズムＰｒｏｖｅ$_{Ｃｏｎｖ}$と、（Ｖ$_{c}$）検証アルゴリズムＶｅｒｉｆｙ$_{Ｃｏｎｖ}$と、で構成される。

（ａ$_{ｉ，ｊ}$）はＣｏｎｖの重みパラメータである。このとき、入力ｘ＝（ｘ$_{１，１}$，・・・，ｘ$_{１，ｍ’}$，・・・，ｘ$_{ｍ，１}$，・・・，ｘ$_{ｍ，ｍ}$）と出力ｙ＝（ｙ$_{１，１}$，・・・，ｙ$_{１，ｍ’}$，・・・，ｙ$_{ｍ，１}$，・・・，ｙ$_{ｍ，ｍ}$）に対して、ｙ$_{ｉ，ｊ}$は以下の式で表せる。

【０１６１】

【数４２】

$$y_{i,j} = \sum_{i'=0}^{s-1} \sum_{j'=0}^{t-1} a_{i,j} x_{i+i',j+j'}$$

$$(1 \leq i \leq m-s+1, 1 \leq j \leq m'-t+1)$$

【０１６２】

そのため、（１０）Ｔｈｅ Ａｆｆｉｎｅ Ｌａｙｅｒプロトコルと同様の方法で、証明Ｐおよび検証結果Ｖを生成することができる。

【０１６３】

【数４３】

$$\mathrm{Prove}_{\mathrm{Conv}}(g, h, x, y, b, A):$$

$$P \coloneqq \mathrm{Prove}_{\mathrm{Affinev}}(g, h, x, y, b, A)$$

$$\mathrm{Verify}_{\mathrm{Conv}}(P):$$

$$V \coloneqq \mathrm{Verify}_{\mathrm{Affine}}(P)$$

【０１６４】

（１２）Ｔｈｅ Ａｖｅｒａｇｅ Ｐｏｏｌｉｎｇ Ｌａｙｅｒプロトコルについて説明する。

Ｔｈｅ Ａｖｅｒａｇｅ Ｐｏｏｌｉｎｇ Ｌａｙｅｒプロトコルは、ｘ、ｙ、ＡＰに関する情報を検証者に与えることなく、ｘ，ｙ∈Ｒ$^{ｍ×ｍ}$に対してｙ＝ＡＰ（ｘ）が成り立つことを証明者が証明するためのプロトコルである。

ＡＰは、畳み込みニューラルネットワークのＡｖｅｒａｇｅ Ｐｏｏｌｉｎｇ層で入力データｘに対して実行される演算である。

Ｔｈｅ Ａｖｅｒａｇｅ Ｐｏｏｌｉｎｇ Ｌａｙｅｒプロトコルは、（Ｐ$_{ｄ}$）証明生成アルゴリズムＰｒｏｖｅ$_{ＡＰ}$と、（Ｖ$_{ｄ}$）検証アルゴリズムＶｅｒｉｆｙ$_{ＡＰ}$と、で構成される。

ｙ，ｘに対して、以下の関係が成立する。

「ｌ」は、ＡＰのフィルタにおいて、行と列のサイズであり、ストライドである。

【０１６５】

【数４４】

$$x = \begin{pmatrix} x_{1,1} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,m} \end{pmatrix}, y = \begin{pmatrix} y_{1,1} & \cdots & y_{1,m} \\ \vdots & \ddots & \vdots \\ y_{m,1} & \cdots & y_{m,m} \end{pmatrix}$$

$$y_{i,j} = \sum_{s=0}^{l-1} \sum_{t=0}^{l-1} \frac{1}{l^2} x_{(l_i+s-1),(l_j+t-1)}$$

$$(1 \le i \le \sqrt{m}, 1 \le j \le \sqrt{m}).$$

【０１６６】
よって、ｙはｘの線形変換の形で表せる。
そのため、（１０）Ｔｈｅ　Ａｆｆｉｎｅ　Ｌａｙｅｒプロトコルと同様の方法で、証明Ｐおよび検証結果Ｖを生成できる。
【０１６７】
【数４５】

$$\text{Prove}_{\text{AP}}(g, h, x, y, l):$$

$$P \coloneqq \text{Prove}_{\text{Affinev}}(g, h, x, y, l)$$

$$\text{Verify}_{\text{AP}}(P):$$

$$V \coloneqq \text{Verify}_{\text{Affine}}(P)$$

【０１６８】
（１３）Ｔｈｅ　Ｍａｘ　Ｐｏｏｌｉｎｇ　Ｌａｙｅｒプロトコルについて説明する。
Ｔｈｅ　Ｍａｘ　Ｐｏｏｌｉｎｇ　Ｌａｙｅｒプロトコルは、ｘ、ｙ、ＭＰに関する情報を検証者に与えることなく、$x \in R^{km \times km}$，$y \in R^{m \times m}$に対してｙ＝ＭＰ（ｘ）が成り立つことを証明者が証明するためのプロトコルである。
ＭＰは、畳み込みニューラルネットワークのＭａｘ　Ｐｏｏｌｉｎｇ層で入力データｘに対して実行される演算である。
Ｔｈｅ　Ｍａｘ　Ｐｏｏｌｉｎｇ　Ｌａｙｅｒプロトコルは、（P_{e}）証明生成アルゴリズムProve_{MP}と、（V_{e}）検証アルゴリズム$\text{Verify}_{\text{MP}}$と、で構成される。
ｙ，ｘに対して、以下の関係が成立する。
「ｋ」は、ＭＰのフィルタにおいて、行と列のサイズであり、ストライドである。
【０１６９】
【数４６】

$$x = \begin{pmatrix} x_{1,1} & \cdots & x_{1,km} \\ \vdots & \ddots & \vdots \\ x_{km,1} & \cdots & x_{km,km} \end{pmatrix}, y = \begin{pmatrix} y_{1,1} & \cdots & y_{1,m} \\ \vdots & \ddots & \vdots \\ y_{m,1} & \cdots & y_{m,m} \end{pmatrix}$$

$$y_{i,j} = \max\left\{ x_{k(i-1)+s,k(j-1)+t} \,\middle|\, s, t \in [1, k] \right\}.$$

【０１７０】
この関係が成立することを証明するために、次の式が成立することを証明する。
【０１７１】
【数４７】

$$\bigwedge_{s=0}^{k}\bigwedge_{t=0}^{k}\left(y_{i,j} \geq x_{k(i-1)+s,k(j-1)+t}\right)$$

$$\wedge \bigvee_{s=0}^{k}\bigvee_{t=0}^{k}\left(y_{i,j} = x_{k(i-1)+s,k(j-1)+t}\right).$$

【０１７２】
$\mathrm{Prove}_{\mathrm{MP}}$は、全ての（ｉ，ｊ）∈［ｍ］×［ｍ］に対して、以下の計算を行う。
【０１７３】
【数４８】

$$\mathrm{Prove}_{\mathrm{MP}}(g,h,x,y,k):$$

$$r_{k(i-1)+s,k(j-1)+t} \leftarrow Z_p \quad (s,t) \in [k]\times[k]$$

$$r'_{i,j} \leftarrow Z_p$$

$$C_{k(i-1)+s,k(j-1)+t} := g^{r_{k(i-1)+s,k(j-1)+t}} h^{\langle x_{k(i-1)+s,k(j-1)+t}\rangle}$$

$$(s,t) \in [k]\times[k]$$

$$C'_{i,j} := g^{r'_{i,j}} h^{\langle y_{i,j}\rangle}$$

【０１７４】
$\mathrm{Prove}_{\mathrm{MP}}$は、（７）Ｒａｎｇｅ　ＰｒｏｏｆｓプロトコルとｎＯＲ　Ｐｒｏｏｆプロトコルを用いて、以下の式に対する証明Ｐを算出する。
【０１７５】
【数４９】

$$\langle y_{i,j}\rangle - \langle x_{k(i-1)+s,k(j-1)+t}\rangle \in \{0,\dots,2^{l+d+1}-1\} \quad (s,t)$$

$$\in [k]\times[k]$$

$$\bigvee_{(s,t)\in[k]\times[k]} C_{k(i-1)+s,k(j-1)+t} / C'_{i,j}$$

$$C'_{i,j} = \delta s,t \ (\delta_{s,t} = r_{k(i-1)+s,k(j-1)+t} - r'_{i,j})$$

【０１７６】
$\mathrm{Verify}_{\mathrm{MP}}$は、$\mathrm{Verify}_{\mathrm{nOR}}$を用いて、検証結果Ｖを生成する。
【０１７７】
【数５０】

$$\mathrm{Verify}_{\mathrm{MP}}(P):$$

$$V \coloneqq \mathrm{Verify}_{\mathrm{nOR}}(P)$$

【０１７８】
（１４）Ｔｈｅ　ＳｏｆｔＭａｘ　Ｌａｙｅｒプロトコルについて説明する。
Ｔｈｅ　ＳｏｆｔＭａｘ　Ｌａｙｅｒプロトコルは、ｘ、ｙに関する情報を検証者に与えることなく、ｘ，ｙ∈R^{n}に対してｙ＝ＳｏｆｔＭａｘ（ｘ）が成り立つことを証明者が証明するためのプロトコルである。
ＳｏｆｔＭａｘは、畳み込みニューラルネットワークのソフトマックス層で入力データｘに対して実行される演算である。
ｘ＝（x_1，・・・，x_{n}），ｙ＝（y_1，・・・，y_{n}）に対して、y_{i}は式（１３Ａ）で表される。
【０１７９】
【数５１】

$$y_i = e^{x_i} / \sum_{j=1}^{n} e^{x_j} \qquad (13A)$$

【０１８０】
Ｔｈｅ　ＳｏｆｔＭａｘ　Ｌａｙｅｒ　Ｐｒｏｔｏｃｏｌは、入力ｘをｘ’＝$\log_2 e^{x}$に置き換えて計算を行う。
よって、次のように式（１３Ａ）に対する証明を構成することができる。ｘ’，ｙ’∈Ｒに対してｙ’＝$2^{x'}$が成立することを証明するプロトコルを構成する。そのプロトコルと（８）Ｍｕｌｔｉｐｌｉｃａｔｉｏｎ　Ｐｒｏｏｆｓプロトコルと合わせることで、証明を構成することができる。
ｙ’＝$2^{x'}$が成立することを証明するプロトコルは、証明生成アルゴリズムＰｒｏｖｅ

ｘｐと、検証結果生成アルゴリズムＶｅｒｉｆｙｅｘｐと、で構成される。
＜ｘ’＞＝ｘ０’2^d＋ｘ１’が成り立つ。本プロトコルは、式（１３Ｂ）を計算する代わりに、８次多項式（１３Ｃ）を用いて、式（１３Ｄ）を計算する。

【０１８１】
【数５２】

$$2^{x_0'+x_1'/2^d} = 2^{x_0'}2^{x_1'/2^d} \qquad (13B)$$

$$P_{1045}(X) = c_8X^8 + \cdots + c_0 \qquad (13C)$$

$$2^{x_1'}P_{1045}(x_0') \qquad (13D)$$

【０１８２】
ただし、ｃ０からｃ８は以下の通りである。

【０１８３】
【数５３】

$$c_0 = 0.10000000000077443021686 \cdot 10^1$$

$$c_1 = 0.69314718042616382779575 6 \cdot 10^0$$

$$c_2 = 0.24022651071017064605384 0 \cdot 10^0$$

$$c_3 = 0.55504068620466379157744 0 \cdot 10^{-1}$$

$$c_4 = 0.96183412258804623749770 0 \cdot 10^{-2}$$

$$c_5 = 0.13327303592814378193290 0 \cdot 10^{-2}$$

$$c_6 = 0.15510746059005257397800 0 \cdot 10^{-3}$$

$$c_7 = 0.14197847399765606711000 0 \cdot 10^{-4}$$

$$c_8 = 0.18633477241379670760000 0 \cdot 10^{-5}$$

【０１８４】
証明生成アルゴリズムＰｒｏｖｅｘｐを以下に記述する。
ｘ０’［ｉ］は、ｘ０’の第ｉビットである。

【０１８５】
【数５４】

$$\mathrm{Prove}_{\exp}(g, h, x'):$$

$$z_1 \coloneqq \langle x_1'^1 \rangle, \ldots, z_8 \coloneqq \langle x_1'^8 \rangle,$$

$$u \coloneqq \left\langle 2^{x_0'} \right\rangle,$$

$$d \coloneqq \langle P_{1045}(x_1') \rangle,$$

$$r_{int}, r_{int}^{(0)}, \ldots, r_{int}^{(l-1)}, r_{real}, r^{(2)}{}_{real}, \ldots, r^{(8)}{}_{real}, r_d, r_u \leftarrow Z_p,$$

【０１８６】
Ｐｒｏｖｅ$_{\exp}$は、（７）Ｒａｎｇｅ　Ｐｒｏｏｆｓプロトコルと（８）Ｍｕｌｔｉｐｌｉｃａｔｉｏｎ　Ｐｒｏｏｆｓプロトコルと（３）一般化Ｓｃｈｎｏｒｒプロトコルを使用して、以下の式に対する証明Ｐを作成する。
【０１８７】
【数５５】

$$C_{int} = g^{r_{int}} h^{x'_0},$$

$$C_{int}^{(i)} = g^{r_{int}^{(i)}} h^{x'_0[i]} \quad i \in \{0, \ldots, l-1\},$$

$$C_{real} = g^{r_{real}} h^{\langle x'_1 \rangle},$$

$$C^{(i)}{}_{real} = g^{r^{(i)}{}_{real}} h^{z_i} \quad i \in \{2, \ldots, 8\},$$

$$C_d = g^{r_d} h^d,$$

$$C_u = g^{r_u} h^u,$$

$$\langle x'_0 \rangle \in \{2^d, \ldots, 2^{d+l-1}\}, \langle x'_1 \rangle \in \{0, \ldots, 2^d - 1\}, \langle x' \rangle = \langle x'_0 \rangle + \langle x'_1 \rangle,$$

$$x_1 = \prod_{i=0}^{l-1} x'_1[i] \cdot 2^i,$$

$$u = \prod_{i=0}^{l-1} \left(x'_1[i] \cdot 2^{2^i} + 1 - x'_1[i] \right),$$

$$z_i = z_{i-1} \cdot \langle x'_1 \rangle, z_1 = \langle x'_0 \rangle \quad i \in \{2, \ldots, 8\},$$

$$d = c_8 \cdot z_8 + \cdots + c_0 = \langle P_{1045}(x'_0) \rangle,$$

$$\langle y \rangle = \langle u \rangle \cdot \langle d \rangle.$$

【０１８８】
検証アルゴリズムＶｅｒｉｆｙ$_{exp}$を以下に記述する。
【０１８９】
【数５６】

$$\text{Verify}_{\text{exp}}(\text{P}):$$

$$V \coloneqq \text{Verify}_{\text{GenShnorr}}(P)$$

【０１９０】
Ｔｈｅ　ＳｏｆｔＭａｘ　Ｌａｙｅｒプロトコルは、（Ｐ$_f$）証明生成アルゴリズムＰｒｏｖｅ$_{SoftMax}$と、（Ｖ$_f$）検証アルゴリズムＶｅｒｉｆｙ$_{SoftMax}$と、で構成される。
【０１９１】

ＰｒｏｖｅＳｏｆｔＭａｘは、Ｐｒｏｖｅｅｘｐ，ＰｒｏｖｅＭｕｌｔを使用して、以下の式に対する証明Ｐを算出する。

【０１９２】

【数５７】

$$\mathrm{Prove}_{\mathrm{SoftMax}}(g,h,x,y):$$

$$y_i{}' = 2^{x_i{}'} \quad i \in \{1,\ldots,n\},$$

$$y_i = y_i'/\sum_{j=1}^{n} y_j'.$$

【０１９３】

ＶｅｒｉｆｙＳｏｆｔＭａｘは、検証結果Ｖを出力する。

【０１９４】

【数５８】

$$\mathrm{Verify}_{\mathrm{SoftMax}}(\mathrm{P}):$$

$$\mathrm{V}{:=}\mathrm{Verify}_{\mathrm{GenShnorr}}(\mathrm{P})$$

【０１９５】

＊＊＊実施の形態１の特徴＊＊＊

実施の形態１は、以下のような特徴を有する。

実施の形態１は、推論モデルのモデルパラメータを整数値化し、推論モデルの検証を行う。

各層のゼロ知識証明プロトコルのアルゴリズムは、実施の形態１の特徴である。

推論モデルの検証手法は、重みパラメータ（モデルパラメータ）の整数値化とゼロ知識証明プロトコルとを組み合わせて実現される。

【０１９６】

カッコ内に符号を記して、実施の形態１の特徴を示す。

推論装置（４００）は、推論処理の対象となるデータ（ｘ）である小数値を整数値で表現し、前記整数値を畳み込みニューラルネットワークのパラメータとして扱って推論モデル（Ｍ）を実行して推論結果（ｃ）を得る。

証明装置（５００）は、前記推論結果を入力にして証明生成アルゴリズムを実行して証明（Ｐ）を得る。

検証装置（６００）は、前記証明を入力にして検証アルゴリズムを実行して検証結果（Ｖ）を得る。

【０１９７】

前記推論結果が、前記畳み込みニューラルネットワークの各層の計算結果を含む。

証明装置（５００）は、前記畳み込みニューラルネットワークの層ごとに、前記層の前記計算結果を入力にして前記層の種類に応じたプロトコルの前記証明生成アルゴリズムを実行する。

前記証明が、前記畳み込みニューラルネットワークの各層に対する前記証明生成アルゴリズムの実行結果を含む。

検証装置（６００）は、前記畳み込みニューラルネットワークの層ごとに、前記層の前

記実行結果を入力にして前記層の種類に応じた前記プロトコルの前記検証アルゴリズムを実行する。

前記検証結果が、前記畳み込みニューラルネットワークの各層に対する前記検証アルゴリズムの実行結果を含む。

【０１９８】

＊＊＊実施の形態１の効果＊＊＊

実施の形態１は、小数の固定小数点表示を整数値で表示することにより、小数のパラメータを扱うことが可能なゼロ知識証明のプロトコルを実現する。

これにより、パラメータが小数である推論モデルに対して、推論モデルの検証を行うことが可能になる。

【０１９９】

実施の形態１は、例えば以下のような効果を奏する。

データが第三者によって分析される。第三者は、機械学習モデルによる推論サービスを提供する者である。

実施の形態１により、分析されたデータの推論結果が実際に機械学習モデルを使ってデータの推論が行われることによって得られた結果であること、を推論モデルに関する情報を開示することなく証明することができる。

【０２００】

実施の形態１の方法は、小数の固定小数点表示を整数値で表示し、離散対数問題の困難性を利用したゼロ知識証明を用いる。これにより、小数のパラメータを扱うことが可能なゼロ知識証明のプロトコルが実現される。そして、パラメータが小数であるような推論モデルに対して、推論モデルの検証を行うことが可能になる。

【０２０１】

＊＊＊実施の形態１の補足＊＊＊

パラメータ生成装置２００と鍵生成装置３００と推論装置４００と証明装置５００は、互いに組み合わされてもよい。つまり、推論検証システム１００は、パラメータ生成装置２００と鍵生成装置３００と推論装置４００と証明装置５００として機能する１台以上のコンピュータを備えてもよい。

【０２０２】

パラメータ生成装置２００の生成部２２０は、公開パラメータｐｐを生成するために、乱数生成機能などを有してもよい。

鍵生成装置３００の生成部３２０は、公開鍵ｐｋと秘密鍵ｓｋを生成するために、乱数生成機能などを有してもよい。

【０２０３】

図１２に基づいて、パラメータ生成装置２００のハードウェア構成を説明する。

パラメータ生成装置２００は処理回路２０９を備える。

処理回路２０９は、受付部２１０と生成部２２０と出力部２３０を実現する処理回路である。

【０２０４】

処理回路は、専用のハードウェアであってもよいし、メモリに格納されるプログラムを実行するプロセッサであってもよい。

処理回路が専用のハードウェアである場合、処理回路は、例えば、単一回路、複合回路、プログラム化したプロセッサ、並列プログラム化したプロセッサ、ＡＳＩＣ、ＦＰＧＡまたはこれらの組み合わせである。

ＡＳＩＣは、Ａｐｐｌｉｃａｔｉｏｎ Ｓｐｅｃｉｆｉｃ Ｉｎｔｅｇｒａｔｅｄ Ｃｉｒｃｕｉｔの略称である。

ＦＰＧＡは、Ｆｉｅｌｄ Ｐｒｏｇｒａｍｍａｂｌｅ Ｇａｔｅ Ａｒｒａｙの略称である。

【０２０５】

パラメータ生成装置２００は、処理回路２０９を代替する複数の処理回路を備えてもよ

い。
【０２０６】
処理回路２０９において、一部の機能が専用のハードウェアで実現されて、残りの機能がソフトウェアまたはファームウェアで実現されてもよい。
【０２０７】
このように、パラメータ生成装置２００の機能はハードウェア、ソフトウェア、ファームウェアまたはこれらの組み合わせで実現することができる。
【０２０８】
図１３に基づいて、鍵生成装置３００のハードウェア構成を説明する。
鍵生成装置３００は処理回路３０９を備える。
処理回路３０９は、受付部３１０と生成部３２０と出力部３３０を実現する処理回路である。
【０２０９】
鍵生成装置３００は、処理回路３０９を代替する複数の処理回路を備えてもよい。
【０２１０】
処理回路３０９において、一部の機能が専用のハードウェアで実現されて、残りの機能がソフトウェアまたはファームウェアで実現されてもよい。
【０２１１】
このように、鍵生成装置３００の機能はハードウェア、ソフトウェア、ファームウェアまたはこれらの組み合わせで実現することができる。
【０２１２】
図１４に基づいて、推論装置４００のハードウェア構成を説明する。
推論装置４００は処理回路４０９を備える。
処理回路４０９は、受付部４１０と推論部４２０と出力部４３０を実現する処理回路である。
【０２１３】
推論装置４００は、処理回路４０９を代替する複数の処理回路を備えてもよい。
【０２１４】
処理回路４０９において、一部の機能が専用のハードウェアで実現されて、残りの機能がソフトウェアまたはファームウェアで実現されてもよい。
【０２１５】
このように、推論装置４００の機能はハードウェア、ソフトウェア、ファームウェアまたはこれらの組み合わせで実現することができる。
【０２１６】
図１５に基づいて、証明装置５００のハードウェア構成を説明する。
証明装置５００は処理回路５０９を備える。
処理回路５０９は、受付部５１０と保管部５２０と証明部５３０と出力部５４０を実現する処理回路である。
【０２１７】
証明装置５００は、処理回路５０９を代替する複数の処理回路を備えてもよい。
【０２１８】
処理回路５０９において、一部の機能が専用のハードウェアで実現されて、残りの機能がソフトウェアまたはファームウェアで実現されてもよい。
【０２１９】
このように、証明装置５００の機能はハードウェア、ソフトウェア、ファームウェアまたはこれらの組み合わせで実現することができる。
【０２２０】
図１６に基づいて、検証装置６００のハードウェア構成を説明する。
検証装置６００は処理回路６０９を備える。
処理回路６０９は、受付部６１０と保管部６２０と検証部６３０と出力部６４０を実現

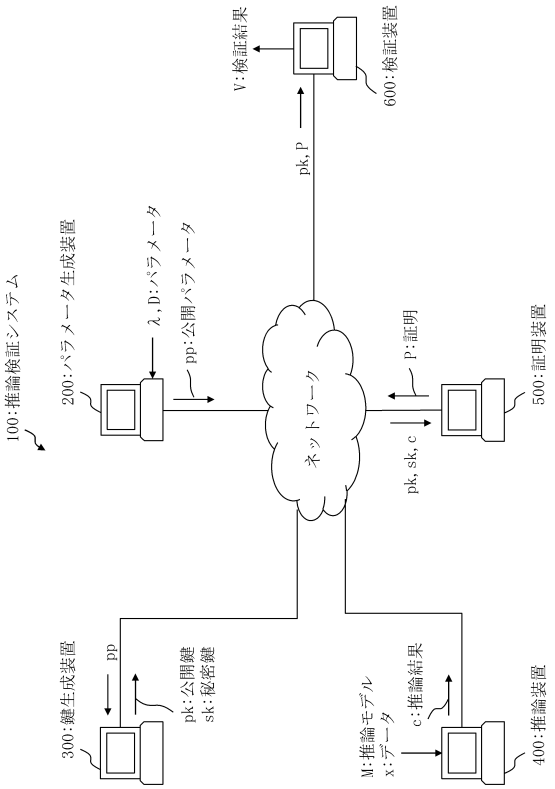
する処理回路である。
【０２２１】
検証装置６００は、処理回路６０９を代替する複数の処理回路を備えてもよい。
【０２２２】
処理回路６０９において、一部の機能が専用のハードウェアで実現されて、残りの機能がソフトウェアまたはファームウェアで実現されてもよい。
【０２２３】
このように、検証装置６００の機能はハードウェア、ソフトウェア、ファームウェアまたはこれらの組み合わせで実現することができる。
【０２２４】
実施の形態１は、好ましい形態の例示であり、本開示の技術的範囲を制限することを意図するものではない。実施の形態１は、部分的に実施してもよいし、他の形態と組み合わせて実施してもよい。フローチャート等を用いて説明した手順は、適宜に変更してもよい。
【０２２５】
推論検証システム１００の各要素の「部」は、「処理」、「工程」、「回路」または「サーキットリ」と読み替えてもよい。
【符号の説明】
【０２２６】
１００ 推論検証システム、２００ パラメータ生成装置、２０１ プロセッサ、２０２ メモリ、２０３ 補助記憶装置、２０４ 通信装置、２０５ 入出力インタフェース、２０９ 処理回路、２１０ 受付部、２２０ 生成部、２３０ 出力部、２９０ 記憶部、３００ 鍵生成装置、３０１ プロセッサ、３０２ メモリ、３０３ 補助記憶装置、３０４ 通信装置、３０５ 入出力インタフェース、３０９ 処理回路、３１０ 受付部、３２０ 生成部、３３０ 出力部、３９０ 記憶部、４００ 推論装置、４０１ プロセッサ、４０２ メモリ、４０３ 補助記憶装置、４０４ 通信装置、４０５ 入出力インタフェース、４０９ 処理回路、４１０ 受付部、４２０ 推論部、４３０ 出力部、４９０ 記憶部、５００ 証明装置、５０１ プロセッサ、５０２ メモリ、５０３ 補助記憶装置、５０４ 通信装置、５０５ 入出力インタフェース、５０９ 処理回路、５１０ 受付部、５２０ 保管部、５２１ 鍵保管部、５２２ 推論結果保管部、５３０ 証明部、５４０ 出力部、５９０ 記憶部、６００ 検証装置、６０１ プロセッサ、６０２ メモリ、６０３ 補助記憶装置、６０４ 通信装置、６０５ 入出力インタフェース、６０９ 処理回路、６１０ 受付部、６２０ 保管部、６２１ 鍵保管部、６２２ 証明保管部、６３０ 検証部、６４０ 出力部、６９０ 記憶部、ｃ 推論結果、Ｄ パラメータ、Ｍ 推論モデル、Ｐ 証明、ｐｋ 公開鍵、ｐｐ 公開パラメータ、ｓｋ 秘密鍵、Ｖ 検証結果、ｘ データ、λ パラメータ。

【要約】
推論装置（４００）は、推論処理の対象となるデータである小数値を整数値で表現し、前記整数値を畳み込みニューラルネットワークのパラメータとして扱って推論モデルを実行して推論結果を得る。証明装置（５００）は、前記推論結果を入力にして証明生成アルゴリズムを実行して証明を得る。検証装置（６００）は、前記証明を入力にして検証アルゴリズムを実行して検証結果を得る。

【図面】

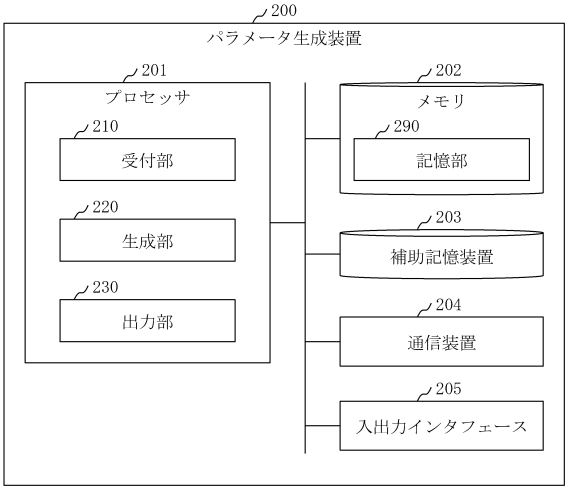
【図１】

図1



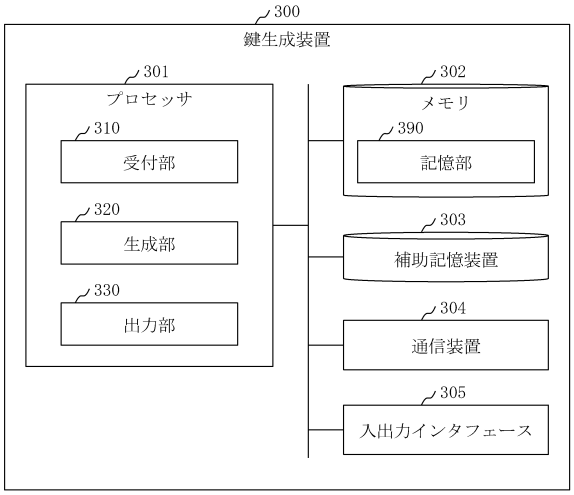
【図２】

図2



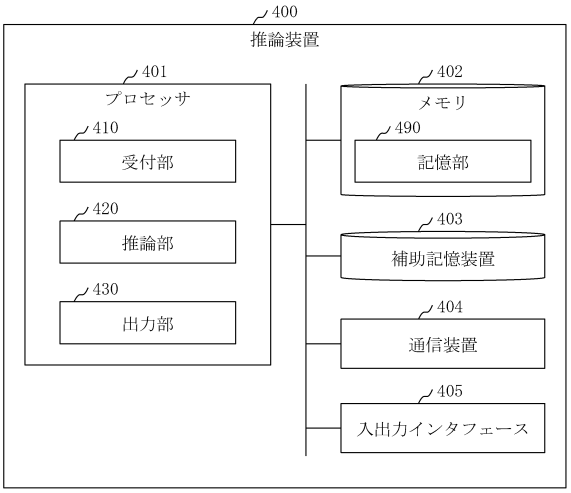
【図３】

図3



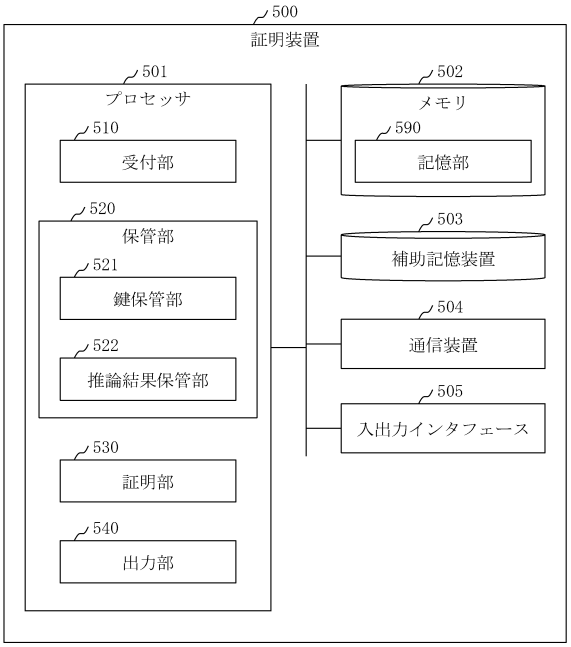
【図４】

図4



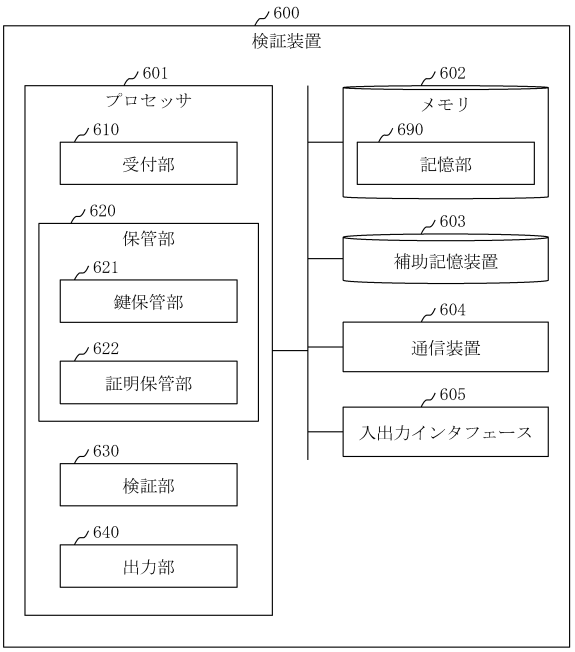
【図５】

図5



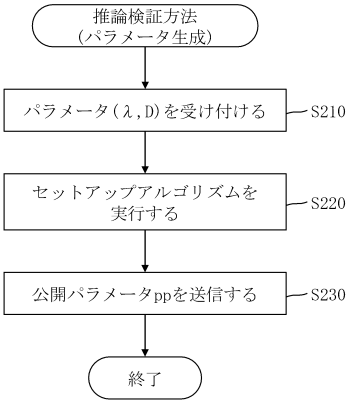
【図６】

図6



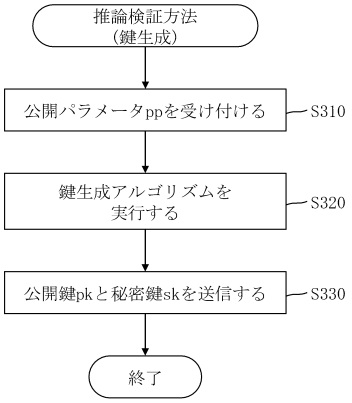
【図７】

図7



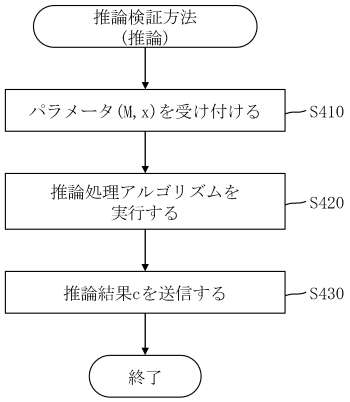
【図８】

図8



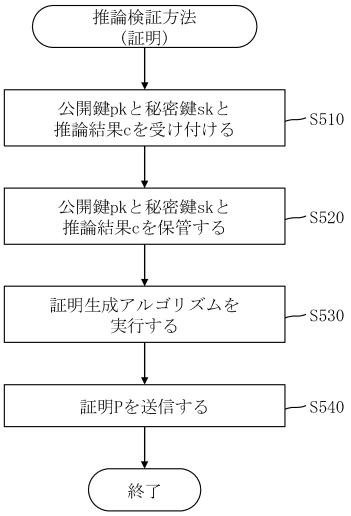
【図９】

図9



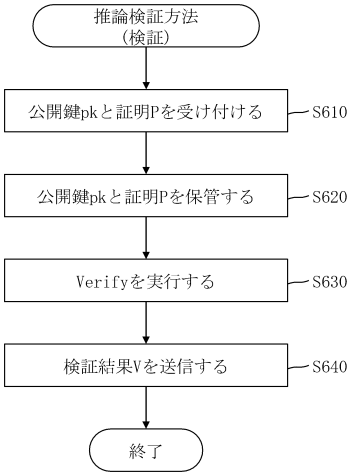
【図１０】

図10



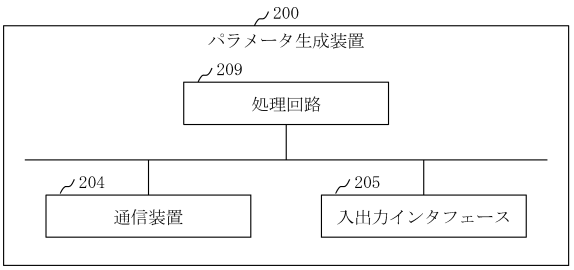
【図１１】

図11



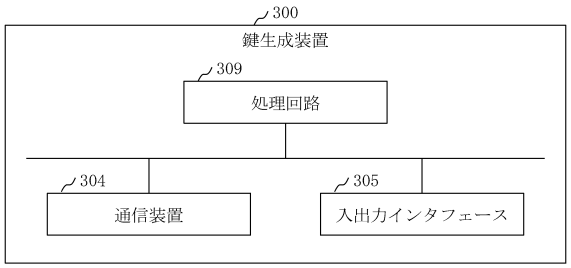
【図１２】

図12



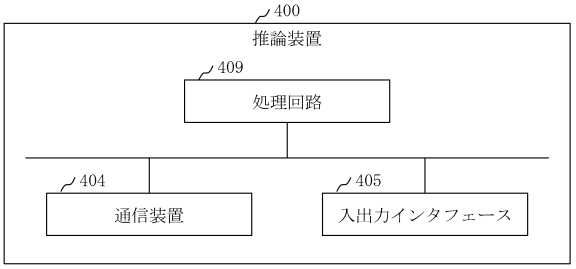
【図１３】

図13



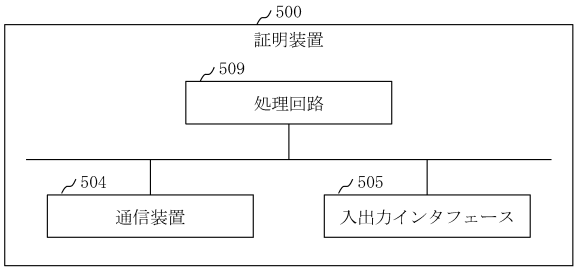
【図１４】

図14



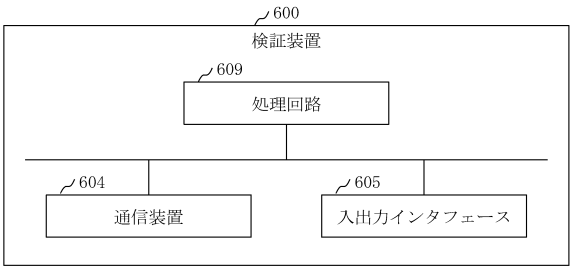
【図１５】

図15



【図１６】

図16



フロントページの続き

東京都江東区青海２－３－２６　国立研究開発法人産業技術総合研究所内
(72)発明者　アッタラパドゥン　ナッタポン
東京都江東区青海２－３－２６　国立研究開発法人産業技術総合研究所内
(72)発明者　坂井　祐介
東京都江東区青海２－３－２６　国立研究開発法人産業技術総合研究所内
(72)発明者　シュルツ　ヤコブ　クロイス　ナカムラ
東京都江東区青海２－３－２６　国立研究開発法人産業技術総合研究所内
審査官　千葉　久博
(56)参考文献　米国特許出願公開第２０２１／０４０６４３６（ＵＳ，Ａ１）
FENG, Boyuan et al.，ZEN: An Optimizing Compiler for Verifiable, Zero-Knowledge Neural Network Inferences，Cryptology ePrint Archive，2021年05月15日，[online], [retrieved on 2022.09.16], Retrieved from the Internet: 〈URL: https://eprint.iacr.org/archive/2021/087/20210515:224258〉
(58)調査した分野　(Int.Cl.，ＤＢ名)
Ｇ０６Ｎ　３／０４６４
Ｇ０６Ｎ　３／０２