学習と推論が可能な確率的演算に基づくエッジ向けAIチップアーキテクチャ

北海道大学大学院情報科学研究院
情報エレクトロニクス部門
教授　浅井 哲也

2021年9月2日

従来技術とその問題点

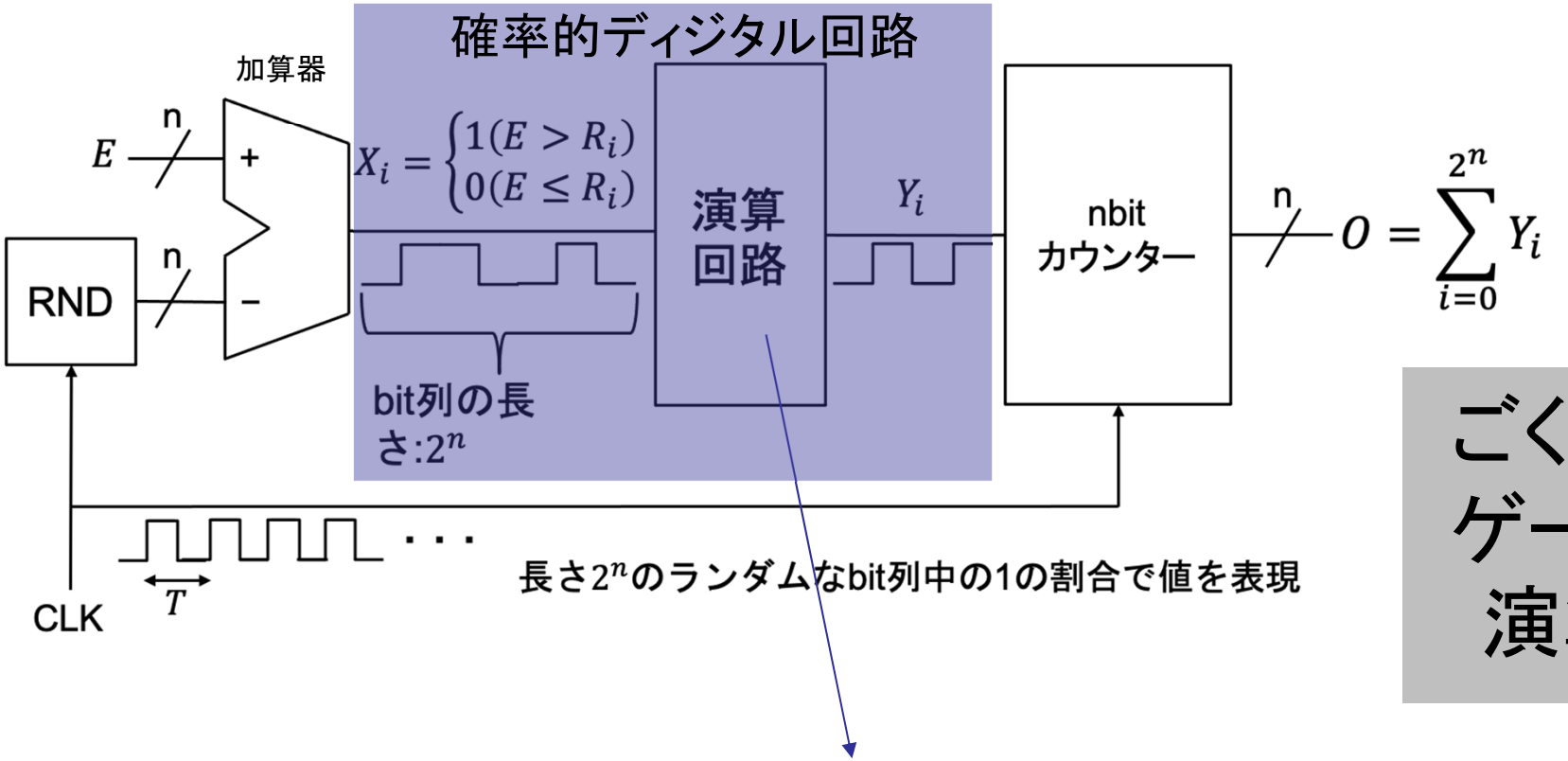
AIチップのアーキテクチャはディジタルが主流であるが、

- 演算器（加算器・乗算器）の並列化によるコスト増
- 消費電力増（ダークシリコン問題）

などの問題により、ディジタルAIチップの電力効率改善は今後より難しくなる方向

本技術は、1960年代に提唱された確率的コンピューティングを再発明し、上記問題の解決を図る。

従来技術：確率的コンピューティング（SC）



ごく僅かな論理ゲートで様々な演算ができる

	乗算	不完全加算	重み付き加算
回路	ANDゲート	ORゲート	マルチプレクサ
論理演算	$Y = X_A * X_B$	$Y = X_A + X_B$	$Y = SX_A + \bar{S}X_B$
確率表記	$p_Y = p_A \bullet p_B$	$p_Y = p_A + p_B - p_A * p_B$	$p_Y = (p_A + p_B)/2$

SCの問題点

確率的コンピューティング（SC）はAIに必要な演算器をディジタル基本ゲートのみで構築可能だが、

- 負の値（減算）の扱いが困難
- 大規模化とともに（AIに必要な）活性化しきい値の設定が困難（スケール問題）
- 活性化関数の演算が困難

といった問題があり（→このあと具体的に説明いたします）、広く利用されるまでには至っていない。

SCを用いたディジタルAIチップの問題点

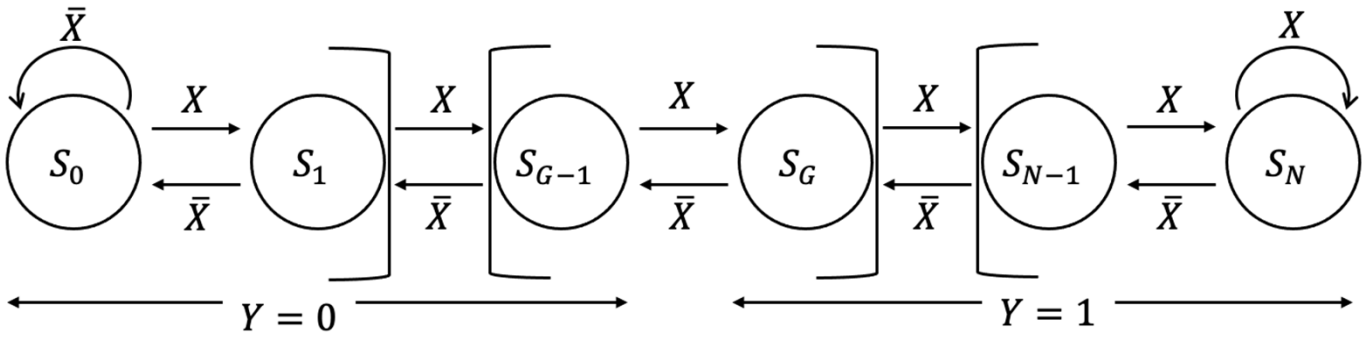
従来技術1.

ステートマシンを用いた活性化関数の実装

(https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=954505&tag=1)

負の値を扱えない

回路規模大・スケール問題

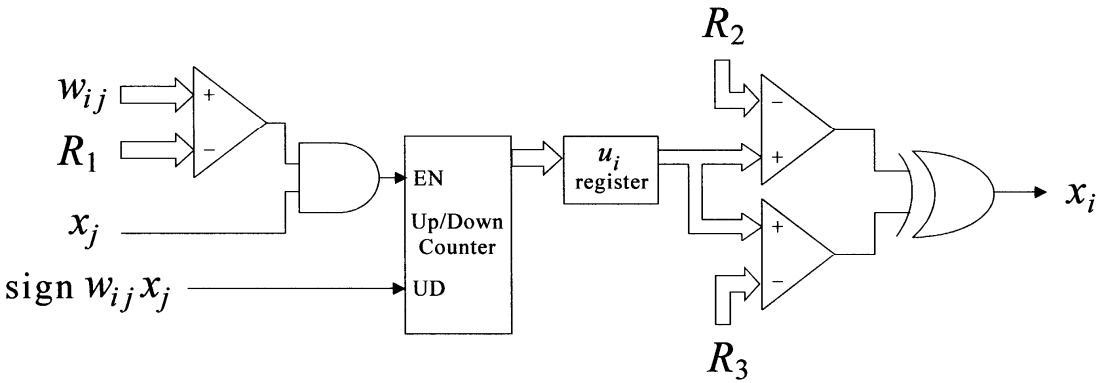


従来技術2.

ディジタルコンパレータを用いた活性化関数

(http://hdl.handle.net/10097/48248)

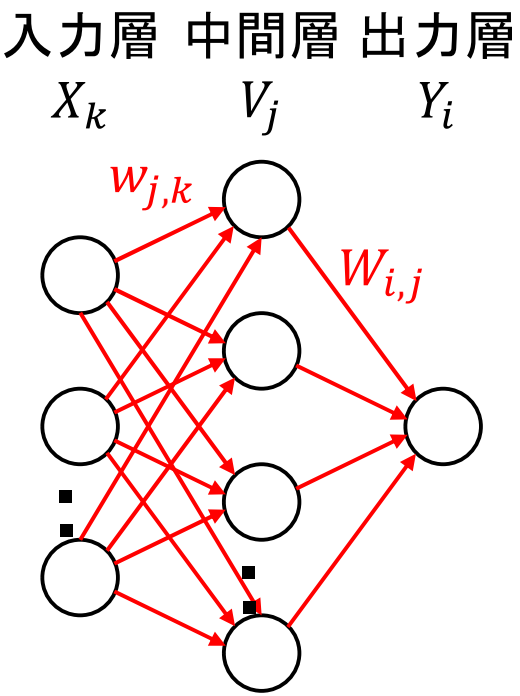
深刻な回路規模増加



新技術の特徴・従来技術との比較

	活性化関数	加減算	
		加算	減算
従来研究1	ステートマシン（規模大）	重み付き加算（スケール問題→デコードが必要→回路規模増）	不可
従来研究2	デコードが必要（深刻な回路規模増）	デコードが必要（深刻な回路規模増）	
新技術	ORゲート・ANDゲートのみ	不完全加算(ORゲートのみ)	シャント抑制（NOT・ANDゲートのみ）

AIに必要な演算の概要



学習手順

1.データを入力して演算を行う(forward)

$$V_j = g(h_{j,k}) = g\left(\sum_{k=0}^{N_X-1} w_{j,k} * X_k\right)$$

$$Y_i = g(h_{i,j}) = g\left(\sum_{j=0}^{N_V-1} W_{i,j} * V_j\right)$$

活性化関数:$y = g(x)$

2.演算結果に基づいて重みを更新(Back Propagation)

確率的勾配降下法によって重みを更新

確率的勾配降下法

誤差関数Eを$E = \frac{1}{2}\sum_{i=0}^{N_Y-1}(Y_i - \mathrm{Z}_i)^2$と定義して、forward演算の入出力に対して誤差関数が極小値に向かうように重みの更新を行う。

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} \quad \Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}}$$

新技術(1):活性化関数

$$p(Y=1) = p_A + p_B - p_A * p_B$$

特に$p_A = p_B$ならば

$$p(Y=1) = 2p_A - p_A^2$$

非線形性をもった出力

$$g(x) = (2x - x^2)^{2n}$$

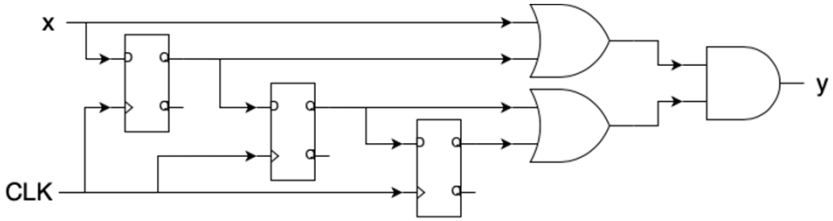
(nは自然数)

入力確率の独立性

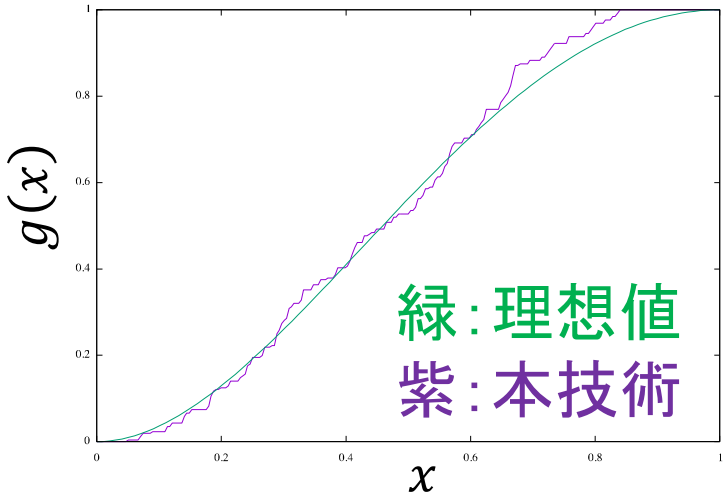
SCでは論理ゲートの入力が**互いに独立**でなければならない
→同じ入力を用いるときは1クロック以上の遅延が必要

$n = 1$の例

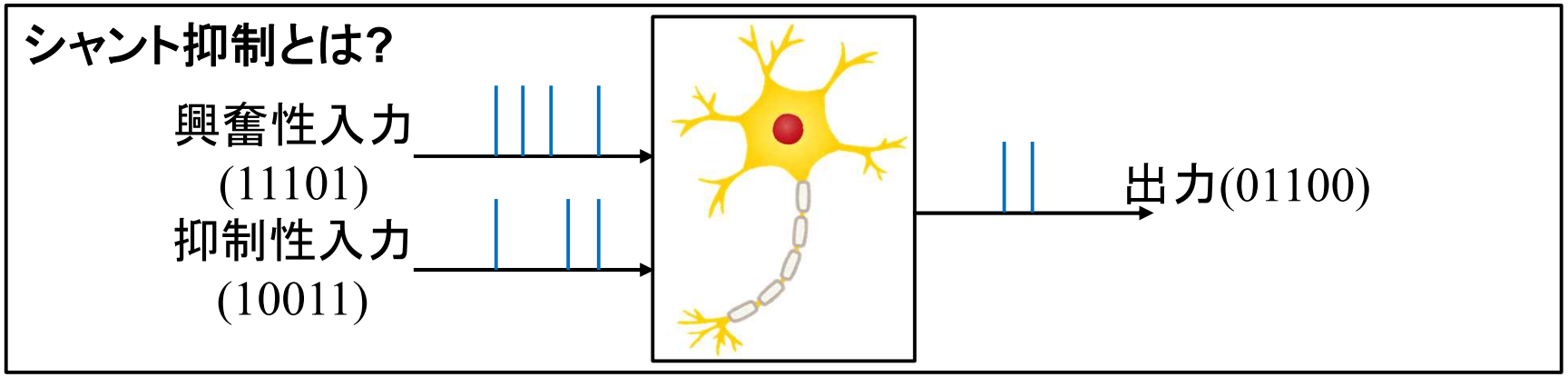
$$g(x) = (2x - x^2)^2$$



活性化関数の理論値と
回路出力の比較



新技術(2):シャント抑制による減算



確率…[0,1]の値しか表せない
重み…正負の値を取り扱う

従来の確率的コンピューティングでは、
差動回路を用いる必要がある

既存手法

$$Y^+ = X_A * \overline{X_B}$$
$$Y^- = \overline{X_A} * X_B$$
$$Y = Y^+ - Y^-$$

回路面積 : **大**
演算精度:**高**

提案手法

$$Y = X_A * \overline{X_B}$$

回路面積 : **小**
演算精度:**低**

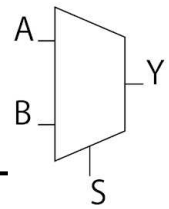
この演算精度の問題をなくすために学習則に手を加える

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} \quad \Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}}$$

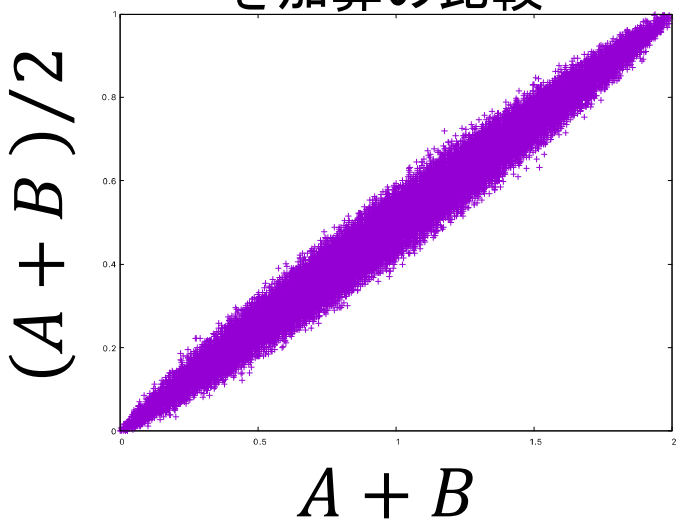
新技術(3):不完全加算と活性化

重み付き加算(既提案)

$$p_Y = (p_A + p_B)/2$$



加算の理論値と重み付き加算の比較

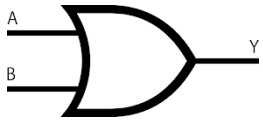


精度は高いが出力が規格化されてしまう

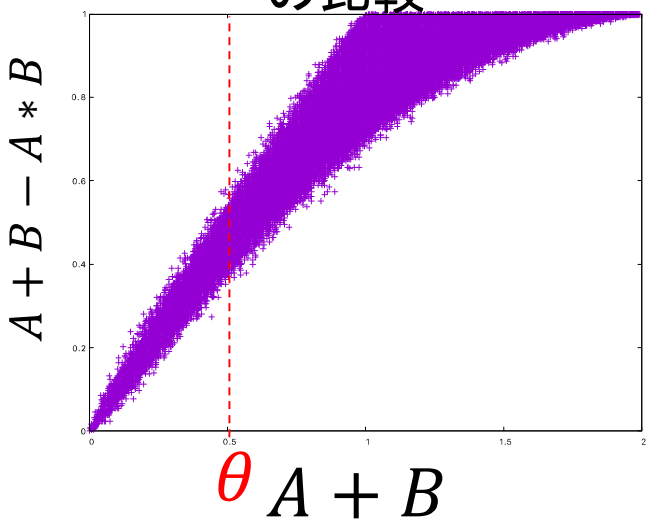
→スケール問題(デコードが必要)

不完全加算(本提案)

$$p_Y = p_A + p_B - p_A * p_B$$

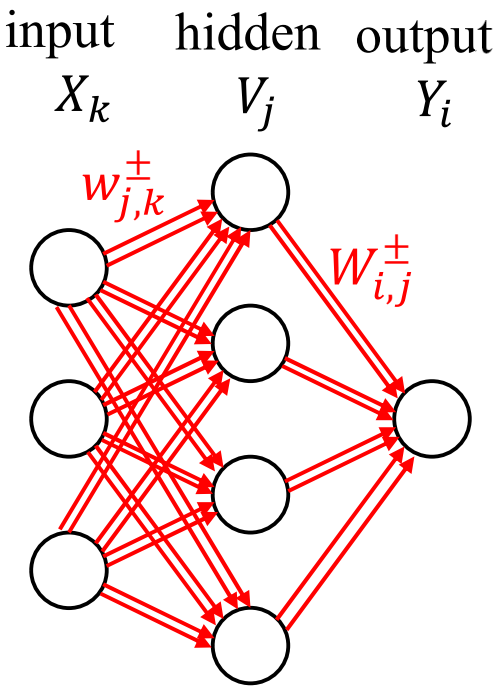


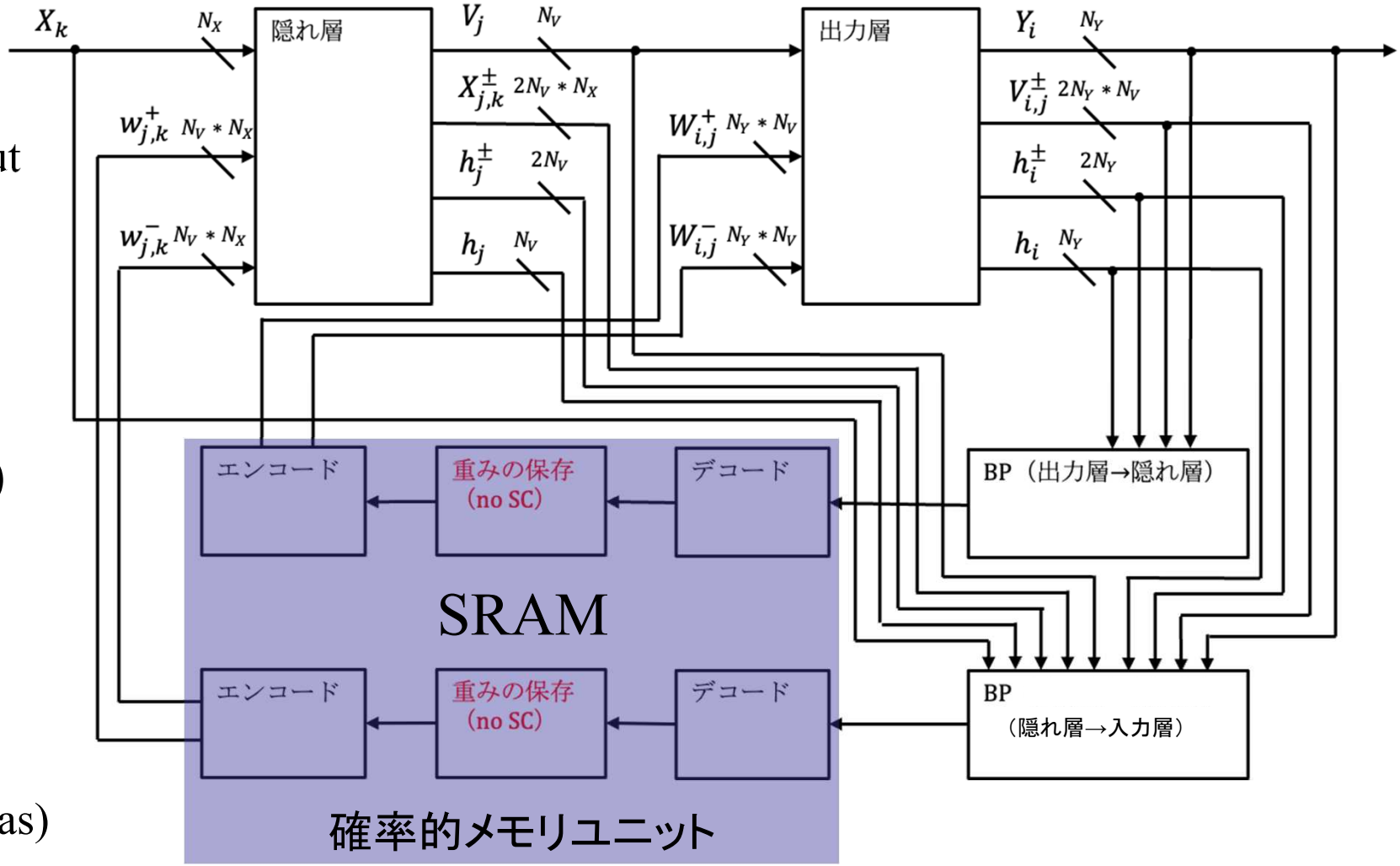
加算の理論値と不完全加算の比較



全体で精度は落ちるが
(1)閾値が小さければ精度は高い
(2)ランダムノイズと定性的に同じ

MLPアーキテクチャ（全体ブロック図）



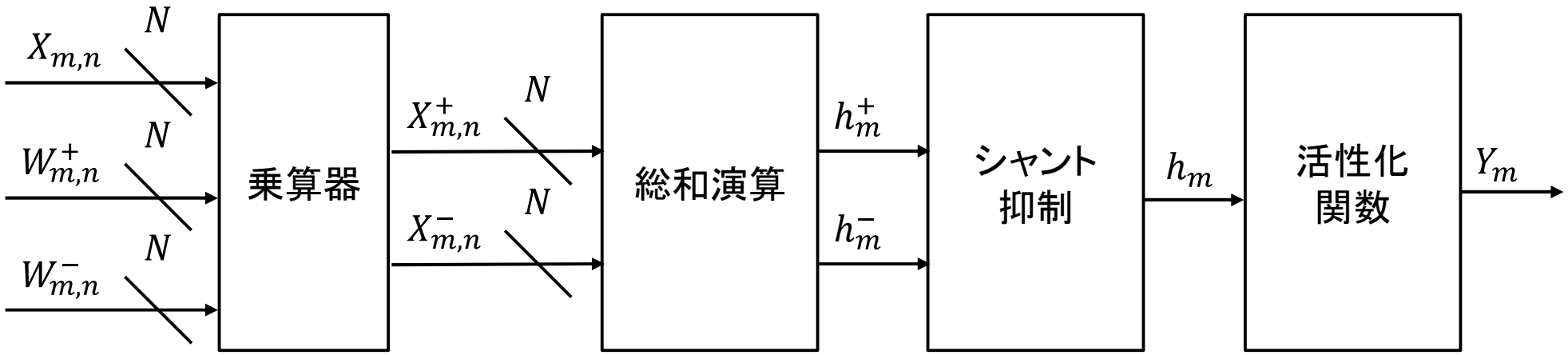


学習パラメータ
入力層 : N_X =2+1(bias)
中間層 : N_V = 4
出力層 : N_Y = 1
学習率 : 0.3
bit列長 : 255

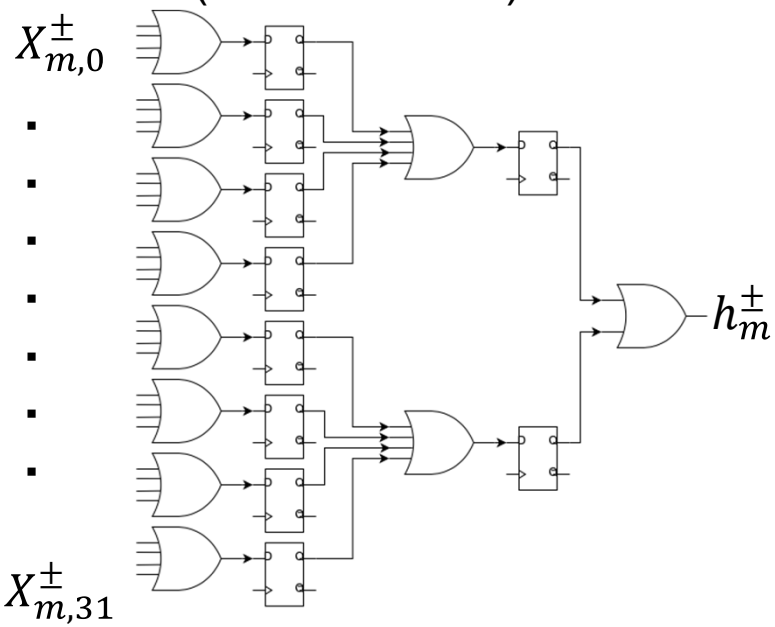
デコード・エンコード→オーバーヘッド大
デコード・エンコードが不要な専用デバイスが必要

ニューロンのアーキテクチャ(ブロック図)

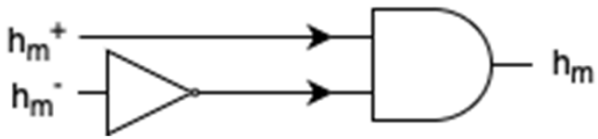
ニューロン単体(N入力1出力)



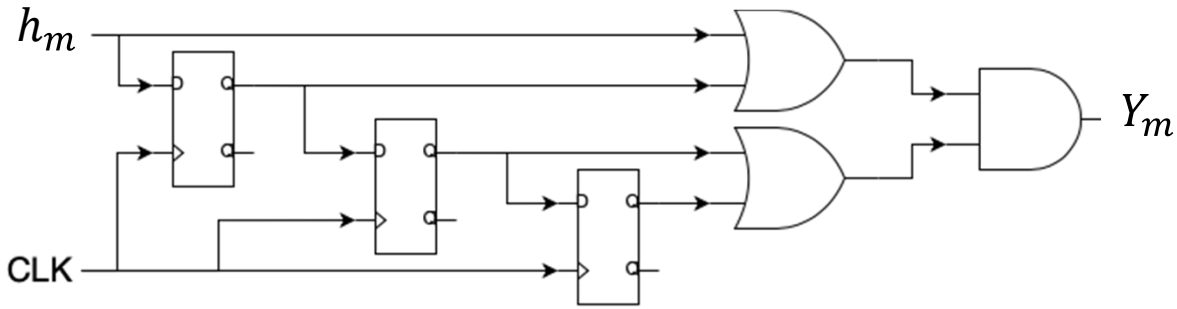
総和演算($N = 32$の場合)



シャント抑制

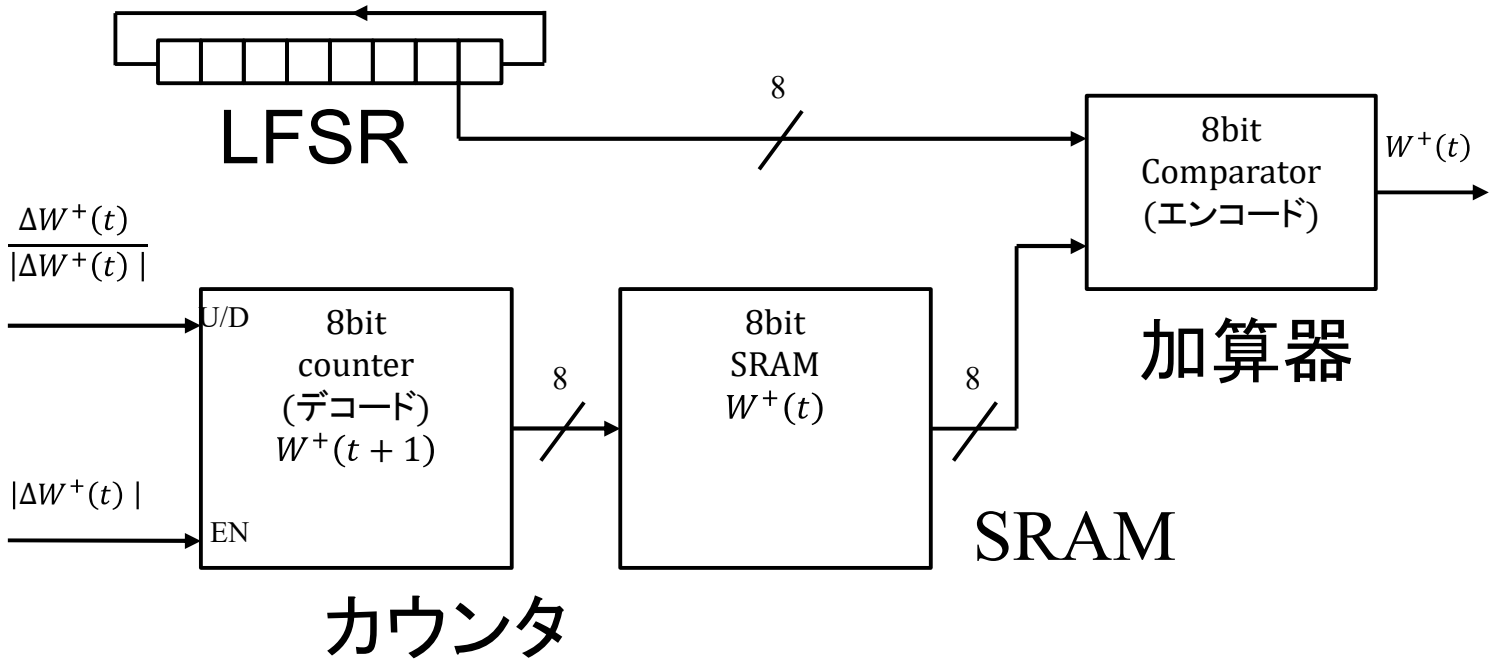


活性化関数



確率的メモリユニット（要最適化）

（エンコード・デコード型：オーバーヘッド大なので改善したい点）



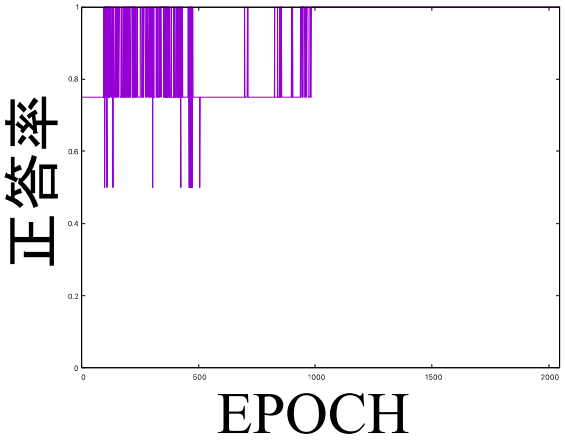
手順

(1)(n+1)bitカウンタでデコード
(2)1EPOCH毎にカウンタの値をレジスタにコピー
(3) n bit LFSRで生成したn bitの乱数列とレジスタの値を比較
(4)比較結果によって確率ストリームを出力

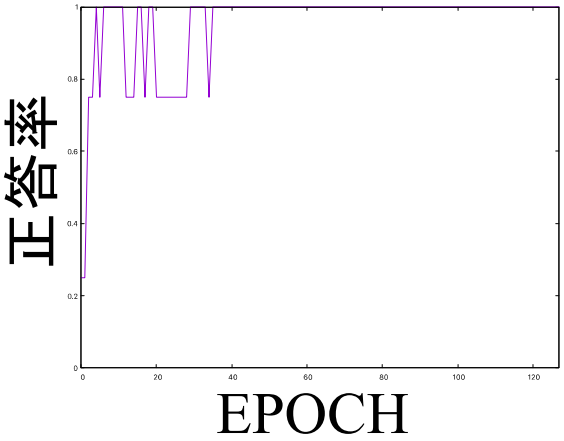
評価結果1（論理関数の学習）

線形分離可能問題

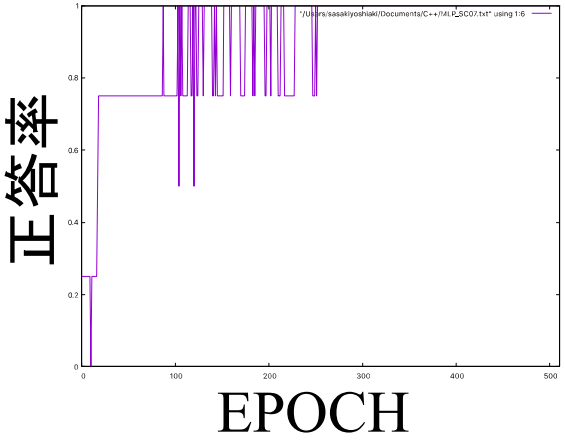
AND



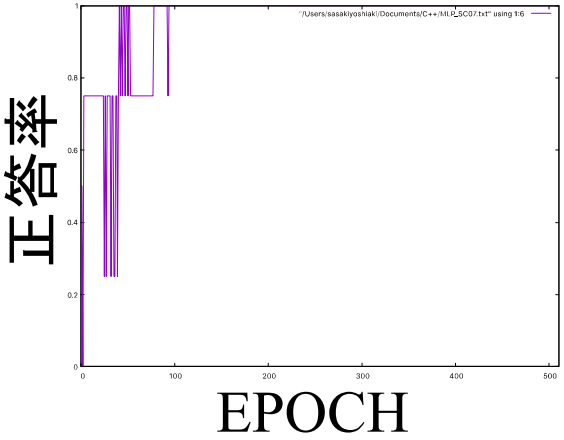
OR



NAND

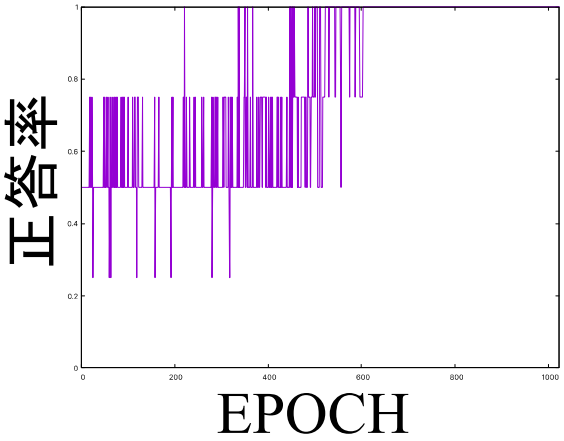


NOR

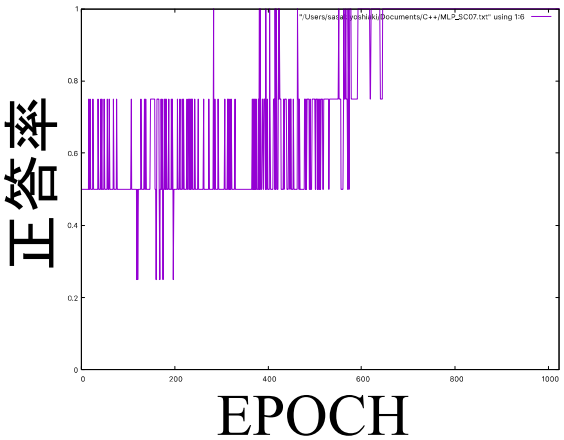


線形分離不可能問題

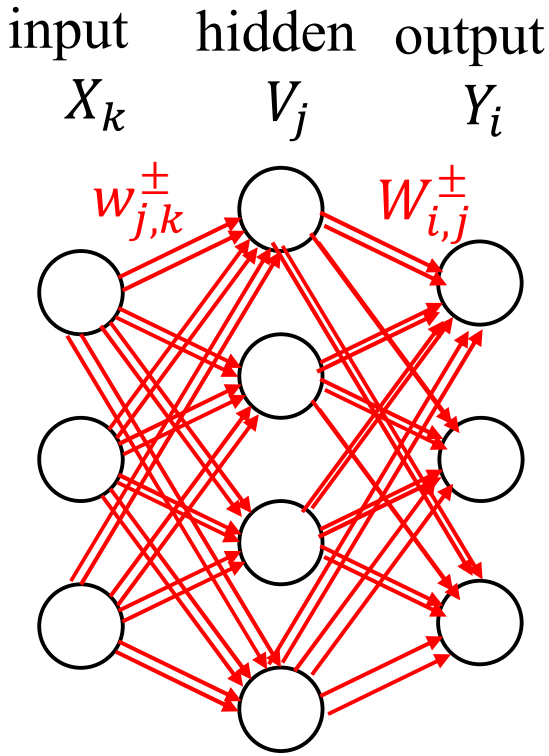
XOR



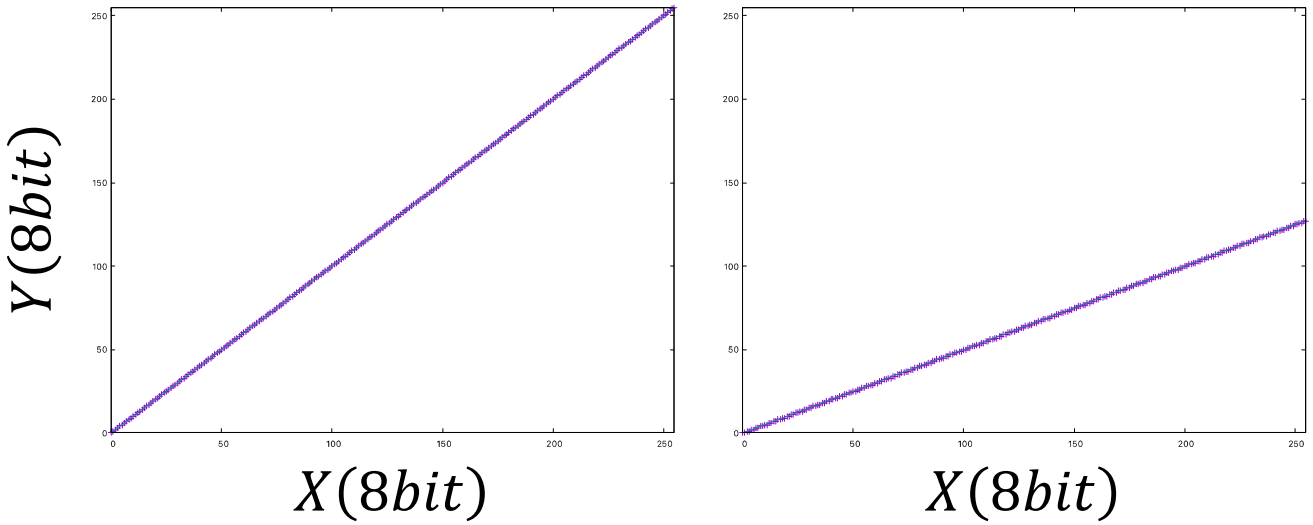
XNOR



評価結果2（線形回帰問題）



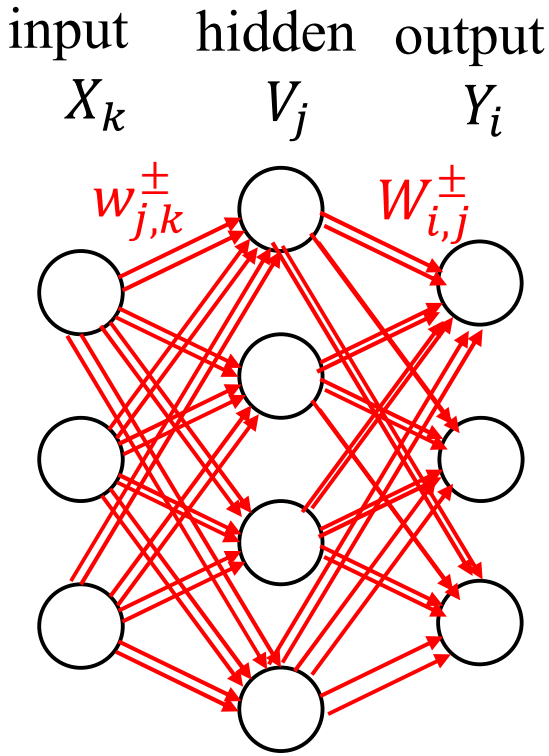
線形回帰問題(左:$y = x$、右: $y = 1/2\,x$)の学習後、
学習した重みを用いて推論を行った結果



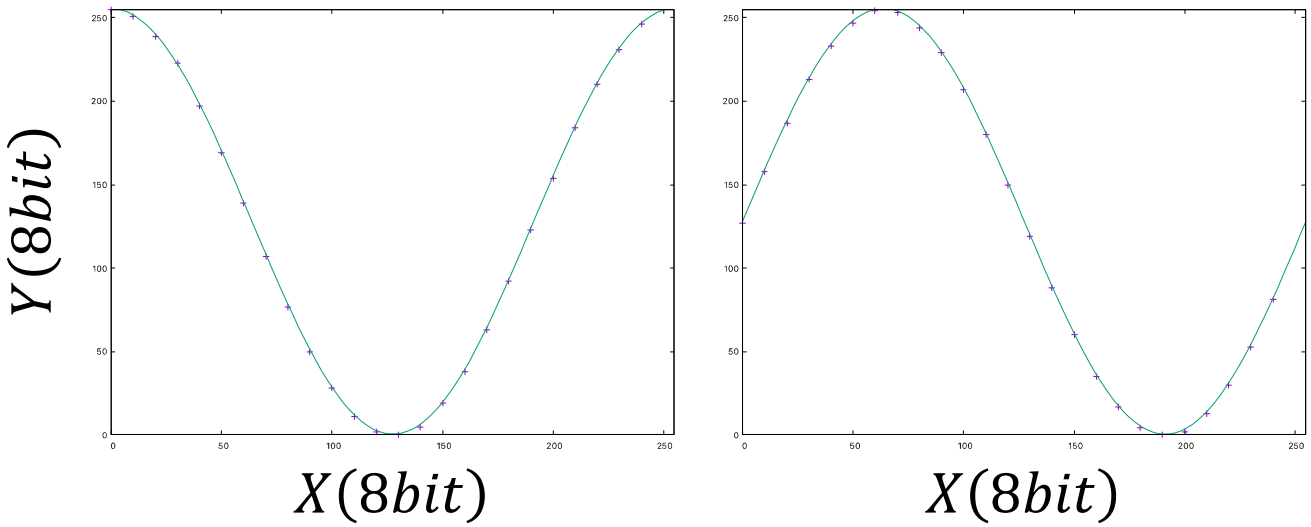
学習パラメータ
入力層 : 8+1(bias)
中間層 : 32
出力層 : 8
学習率 : 0.3
bit列長 : 255

教師データ
入力 : $x = n$(nは0以上の整数)
出力 : $y = x$、$y = 1/2\,x$

評価結果3（非線形回帰問題）



非線形回帰問題(左:cos(x)、右:sin(x))の学習後、
学習した重みを用いて推論を行った結果



学習パラメータ
入力層 : 8+1(bias)
中間層 : 128
出力層 : 8
学習率 : 0.3

教師データ
入力 : $x = 10n$(nは0以上の整数)
出力 : $y = cosx$、$y = sinx$

想定される用途

- エッジにおいて、推論だけでなく「その場学習」が必要となるアプリ（転移学習による個人適応など）
- 上記以外に、個人情報保護の効果（AIパラメータがクラウドで共有されない）が得られることも期待される。
- 真の確率的メモリユニット（真性乱数+不揮発メモリ）の実現により、上記効果がより顕著に。

実用化に向けた課題

- 現在、線形分離不可能問題や非線形回帰問題のオンライン学習が可能なところまで開発済み。しかし、エンコード・デコードが不要な真の確率的メモリの最適化が未解決である。
- 今後、確率的メモリの最適化と消費電力に関する詳細評価を行い、極低電力エッジAI応用に適用していく場合の条件設定を行っていく。
- 実用化に向けて、推論精度を99%程度まで向上できるよう技術を確立する必要もあり。

企業への期待

- 未解決の「確率的メモリの開発」については、TRNG（真性乱数）デバイス・回路と不揮発メモリ技術により克服できる見込である。
- TRNGまたはアナログ不揮発メモリの技術を持つ、企業との共同研究を希望。
- また、組み込みAIデバイスを開発中の企業、その場学習が可能なエッジAI分野への展開を考えている企業には、本技術の導入が有効と思われる。

本技術に関する知的財産権

- 発明の名称　:小型極低電力学習推論装置
- 出願番号　:特願2021-118326
- 出願人　:北海道大学
- 発明者　:浅井哲也、佐々木義明、西田浩平

産学連携の経歴

- 2001-2001年 NEDO 即効型産業技術研究助成事業に採択
- 2005-2006年 NEDO 産業技術研究助成事業に採択
- 2016-2021年 NEDO「IoT推進のための横断技術開発プロジェクト」事業に採択
- 2018-2023年 NEDO「高効率・高速処理を可能とするAIチップ・次世代コンピューティングの技術開発」事業に採択
- 民間企業との共同研究実績：7件

お問い合わせ先

北海道大学
産学・地域協働推進機構　岩村相哲

ＴＥＬ 011-706-9559
ＦＡＸ 011-706-9550
e-mail s-iwamura@mcip.hokudai.ac.jp

以下、補足スライド

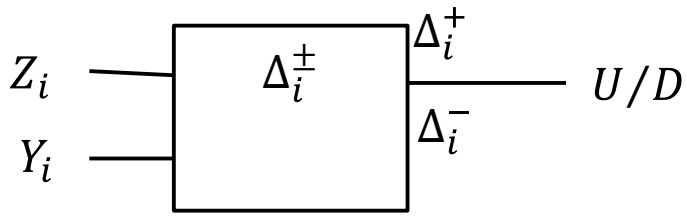
BPブロック及び回路(出力層→隠れ層)

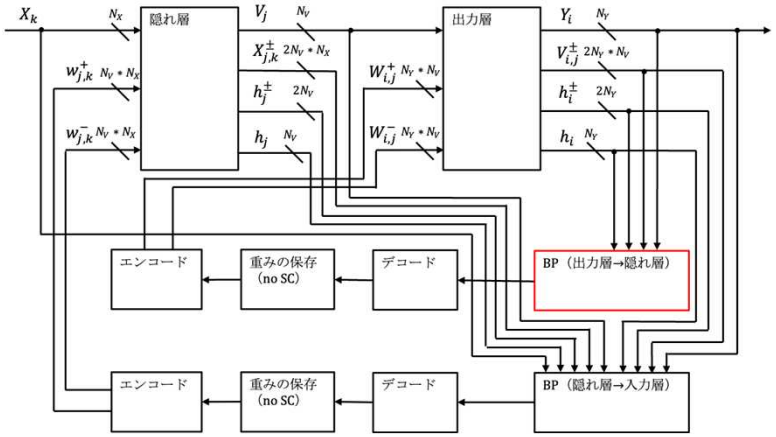
$$\Delta W_{ij}^{+} = -\eta \frac{dE}{dW_{ij}^{+}} = \eta * (Z_i - Y_i) * g'\left(h_{Y_i}^{+} * \left(1 - h_{Y_i}^{-}\right)\right) * \left(1 - h_{Y_i}^{-}\right) * V_j * \left(1 - \sum_{n \neq j}^{OR} W_{i,n}^{+} * V_n\right)$$

$0 \leq n \leq N_V - 1$のうち$n \neq j$を満たすものの総和

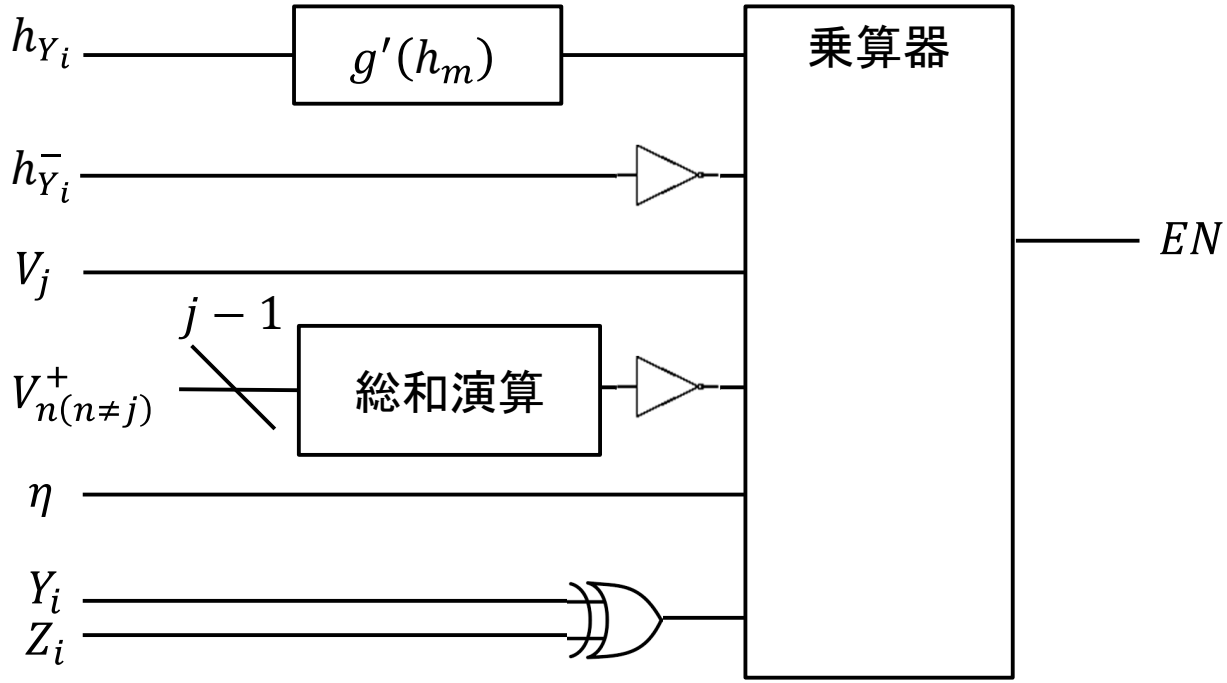
符号決定回路

（確率的メモリユニットのオーバーヘッド部分）



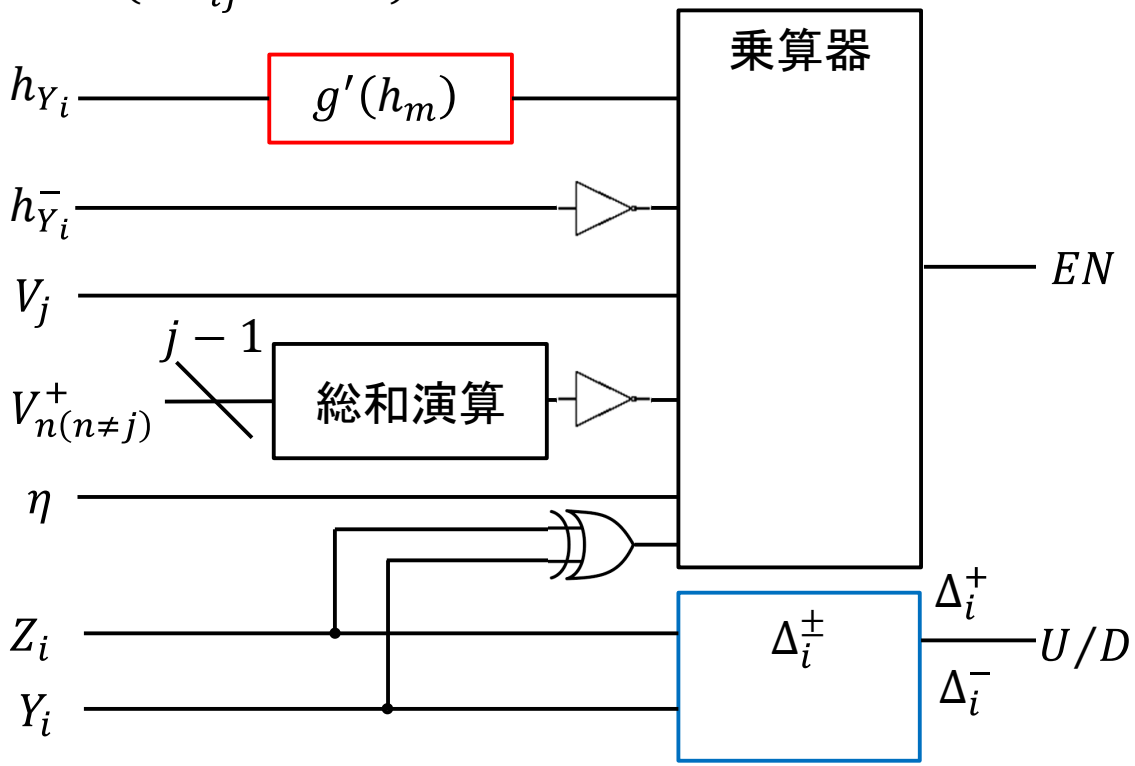


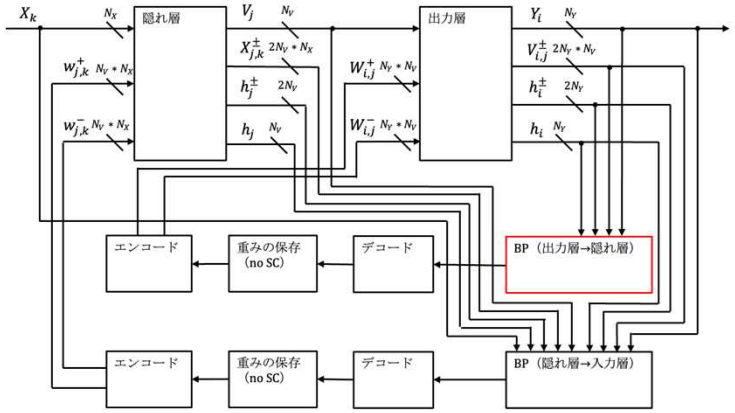
絶対値決定回路



BPブロック及び回路(出力層→隠れ層)

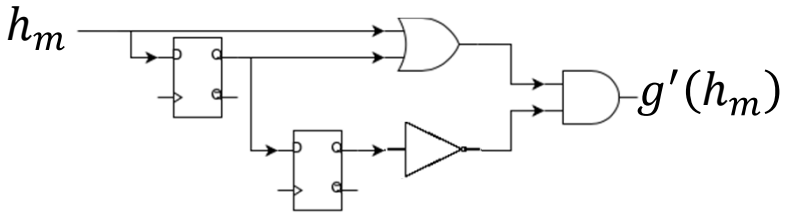
ブロック図(ΔW_{ij}^{+}の場合)



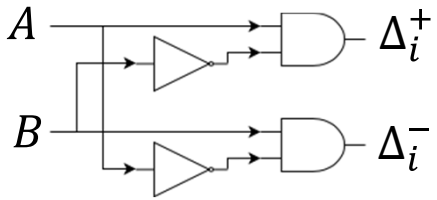


（確率的メモリユニットのオーバーヘッド部分）

活性化関数の微分($g'(h_m)$)



$A-B$の符号判定回路($\Delta_i^{\pm}$)



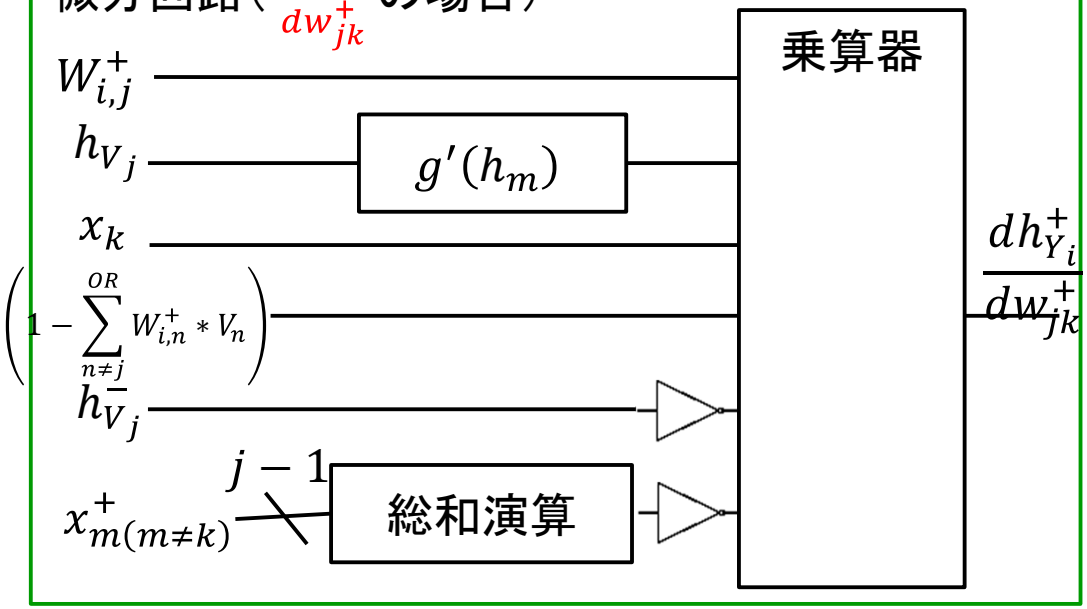
BPブロック及び回路(隠れ層→入力層)

$$\Delta w_{jk}^{+} = -\eta \frac{dE}{dw_{jk}^{+}} = \frac{1}{N_Y} \sum_{i=0}^{N_Y-1} \eta * (Z_i - Y_i) * \left(\left(1 - h_{Y_i}^{-}\right) \frac{dh_{Y_i}^{+}}{dw_{jk}^{+}} - h_{Y_i}^{+} \frac{dh_{Y_i}^{-}}{dw_{jk}^{+}} \right) * g'\left(h_{Y_i}^{+} * \left(1 - h_{Y_i}^{-}\right)\right)$$

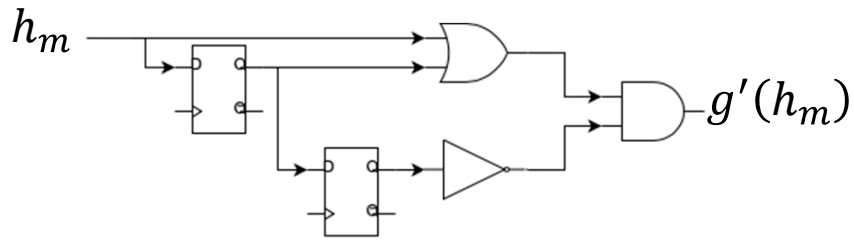
$$\frac{dh_{Y_i}^{+}}{dw_{jk}^{+}} = W_{i,j}^{+} * g'\left(h_{V_j}\right) * x_k * \left(1 - \sum_{n \neq j}^{OR} W_{i,n}^{+} * V_n\right) * \left(1 - h_{V_j}^{-}\right) * \left(1 - \sum_{m \neq k}^{OR} w_{i,j}^{+} * x_m\right)$$

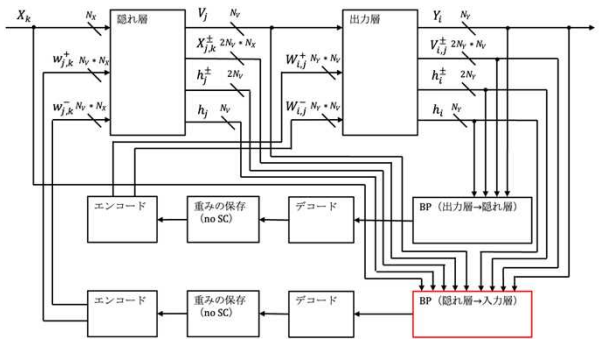
$$\frac{dh_{Y_i}^{-}}{dw_{jk}^{+}} = W_{i,j}^{-} * g'\left(h_{V_j}\right) * x_k * \left(1 - \sum_{n \neq j}^{OR} W_{i,n}^{-} * V_n\right) * \left(1 - h_{V_j}^{-}\right) * \left(1 - \sum_{m \neq k}^{OR} w_{i,j}^{+} * x_m\right)$$

微分回路（$\frac{dh_{Y_i}^{+}}{dw_{jk}^{+}}$ の場合）



活性化関数の微分（$g'(h_m)$）

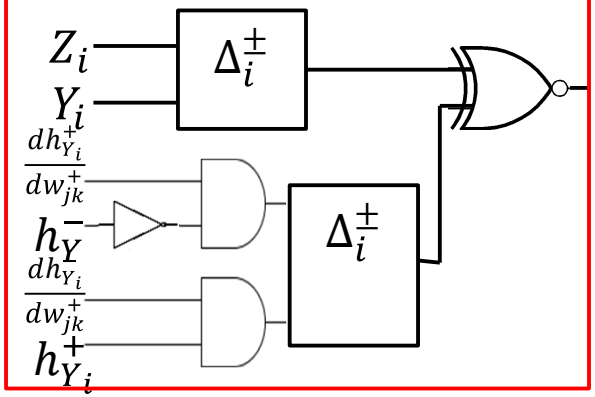


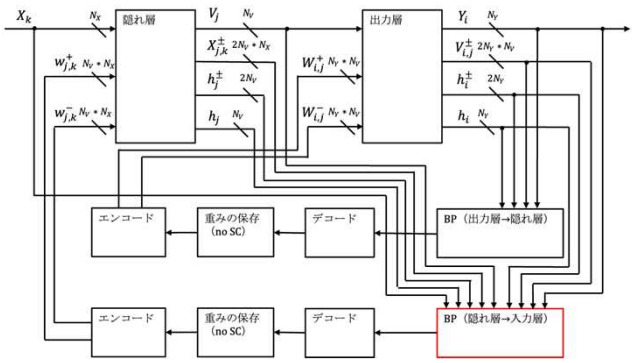


BPブロック及び回路(隠れ層→入力層)

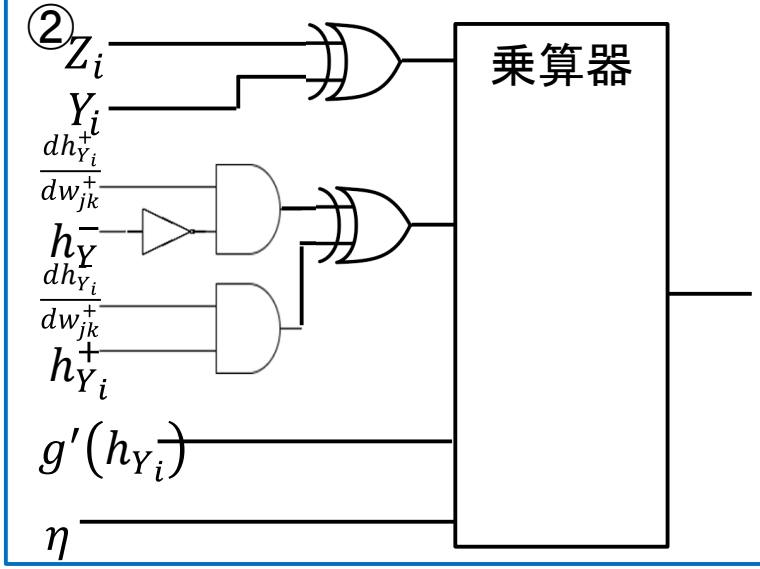
$$\Delta w_{jk}^{+} = -\eta \frac{dE}{dw_{jk}^{+}} = \frac{1}{N_Y}\sum_{i=0}^{N_Y-1} \eta * (Z_i - Y_i) * \left(\left(1 - h_{Y_i}^{-}\right) \frac{dh_{Y_i}^{+}}{dw_{jk}^{+}} - h_{Y_i}^{+} \frac{dh_{Y_i}^{-}}{dw_{jk}^{+}} \right) * g'\left(h_{Y_i}^{+} * \left(1 - h_{Y_i}^{-}\right)\right)$$

各出力に対応する
符号決定回路①

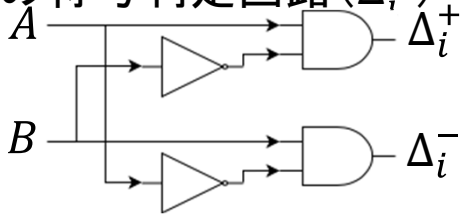




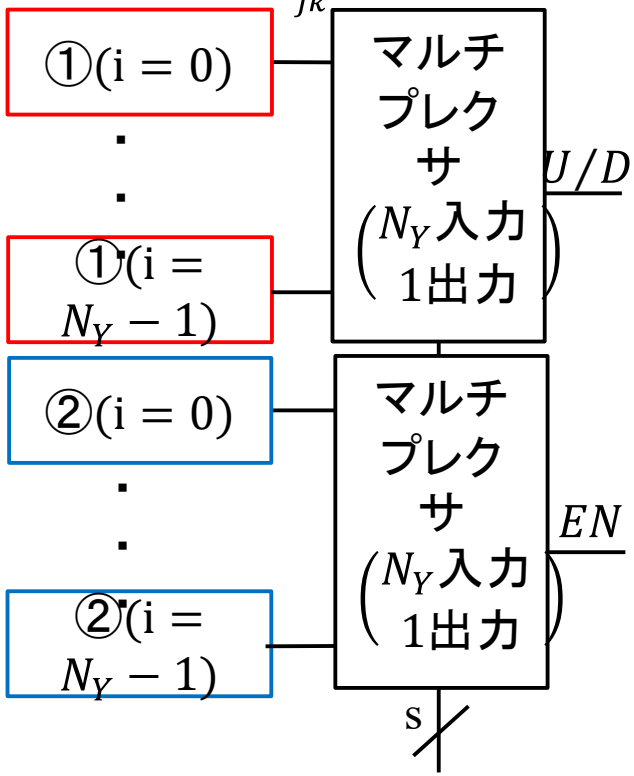
各出力に対応する絶対値決定回路
②



$A - B$の符号判定回路($\Delta_i^{\pm}$)



ブロック図($\frac{dE}{dw_{jk}^{+}}$の場合)



$s = log_2 N_Y$(N_Yが偶数)、$s = log_2(N_Y + 1)$(N_Yが奇数)

確率的メモリユニットにおけるデコード

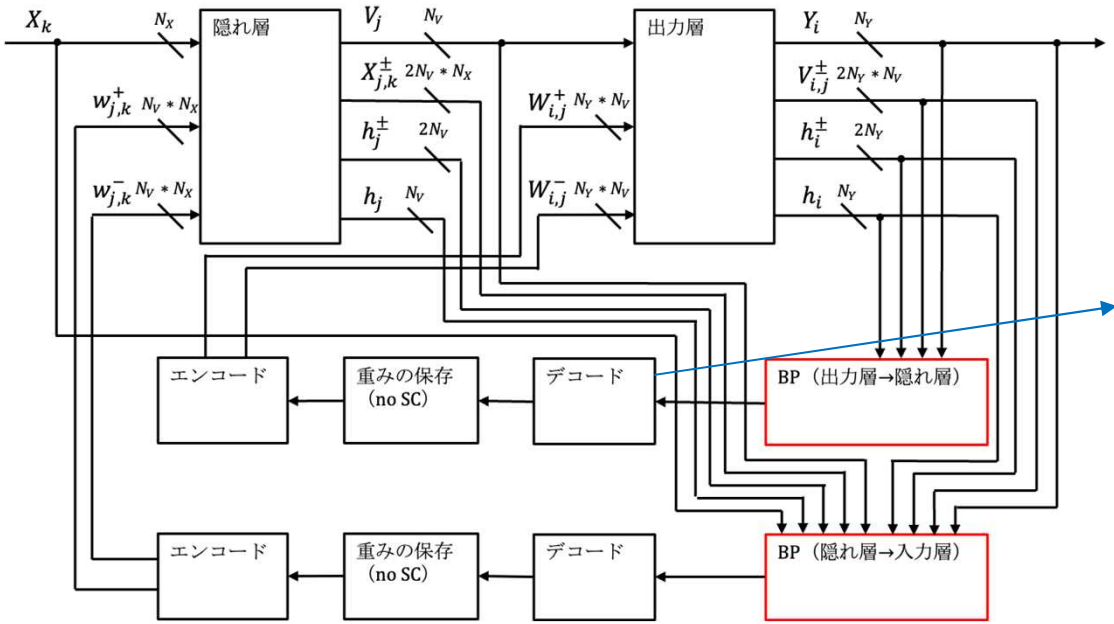
誤差関数 $E = \frac{1}{2}\sum_{i=0}^{N_Y-1}(Y_i - Z_i)^2$

オーバーヘッド大なので改善したい

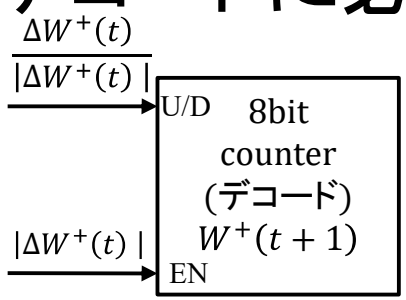
ただし

$$Y_i = g(h_{i,j}) = g\left(\left[\sum_{j=0\to N_V-1}^{OR} W_{i,j}^{+} * V_j\right] * \left[1 - \sum_{j=0\to N_V-1}^{OR} W_{i,j}^{-} * V_j\right]\right)$$

$$V_j = g(h_{j,k}) = g\left(\left[\sum_{k=0\to N_X-1}^{OR} w_{j,k}^{+} * X_k\right] * \left[1 - \sum_{k=0\to N_X-1}^{OR} w_{j,k}^{-} * X_k\right]\right)$$



デコードに必要な回路



正のパルス	負のパルス	U/D	EN
0	0	✖	0
1	0	1	1
0	1	0	1
1	1	✖	0

ツーラインのパルス列の状態は3値(1,0,-1)で表現することができるため、UDカウンターでデコードするために必要な入力はパルス列の状態の符号と絶対値のみである。