Gokaku version 1.2

第 35 回世界コンピュータ将棋選手権 アピール文章

1　コンピュータ将棋プログラム「Gokaku」の概要

Gokaku は深層学習を用いたコンピュータ将棋プログラムであり、多くの深層学習を用いたゲーム人工知能のプログラムと同様に、ニューラルネットワークとモンテカルロ木探索を組み合わせて形勢判断を行う。Gokaku は 2025 年 1 月末より開発を開始したコンピュータ将棋プログラムであり、開発期間を短縮するため、将棋の盤面管理、合法手の生成、詰み探索、終局判定には「cshogi」を利用し、盤面評価と候補手探索にはコンピュータ囲碁プログラム「Maru[1)]」のコードを流用している。コンピュータ囲碁プログラム「Maru」は LeelaZero ELF と同等以上の（人間より強い）棋力を持つ強豪プログラムであり、これと同じ技術と用いることで強豪のコンピュータ将棋プログラムを実現することが本開発の目標である。

Gokaku で用いるニューラルネットワークは、7 × 7 のカーネルを用いた Convolution [1] と Multi-Head Attention [2] を含んでおり、盤面全体の情報を効率的に集約できる構造となっている。また、これらのConvolutionやMulti-Head Attentionには表現力を改善するための工夫を施しており、既存のニューラルネットワークよりも複雑な盤面の状態にも対応できるようになっている。さらに、これらの Convolution や Multi-Head Attention は独自に開発したSkip Connection構造 [3]により接続されており、通常の Residual Connection よりも正確に盤面評価を行うことができる。

ニューラルネットワークの学習では、人間の棋譜や既存の将棋 AI の対戦棋譜を用いず、ランダム着手によって生成した棋譜から強化学習を通じて生成した棋譜を学習データとして用いている。強化学習では、AlphaGo Zero [4] と同様に、ニューラルネットワークの自己対戦による棋譜生成と生成した棋譜によるニューラルネットワークの学習を繰り返すことで、より強力なニューラルネットワークを生成することを目指している。ただし、この強化学習では、Katago [5] や Gumbel AlphaZero [6] の知見を取り入れることにより、AlphaGo Zeroよりも効率的に強化学習を行えるように工夫している。

Gokaku の着手決定処理では、多くの深層学習を用いたゲーム人工知能と同様に、PUCT（Polynomial Upper Confidence Tree）に基づくモンテカルロ探索を用いている。ただし、最終的な着手決定では、モンテカルロ木探索で訪問数が最も多くなったノードを選択するのではなく、それぞれのノードの予測勝率の LCB（Lower Confidence Bound）を推定し、LCB の値が最も高いノードを選択することで、少ない探索数の場合でも正確に着手できるように工夫している。また、探索ノードのすべてにおいて 5 手詰めまでの詰み探索を行い、浅い探索ノードにおいては df-pn アルゴリズムによる長手数の詰み探索を行うため、詰みや必死などを考慮した着手決定が可能となっている。

一方、Gokaku には定跡データベースを搭載しておらず、ニューラルネットワークによる盤面評価のみを指し手決定の基準としている。これは、十分に学習を行ったニューラルネットワークであれば定跡データベースと同等の機能を発揮すると考えているため、定跡データベースの搭載よりもニューラルネットワークの性能向上を優先しているためである。

1）　Gokaku と Maru はともに武田敦志が単独で開発しているソフトウェアである

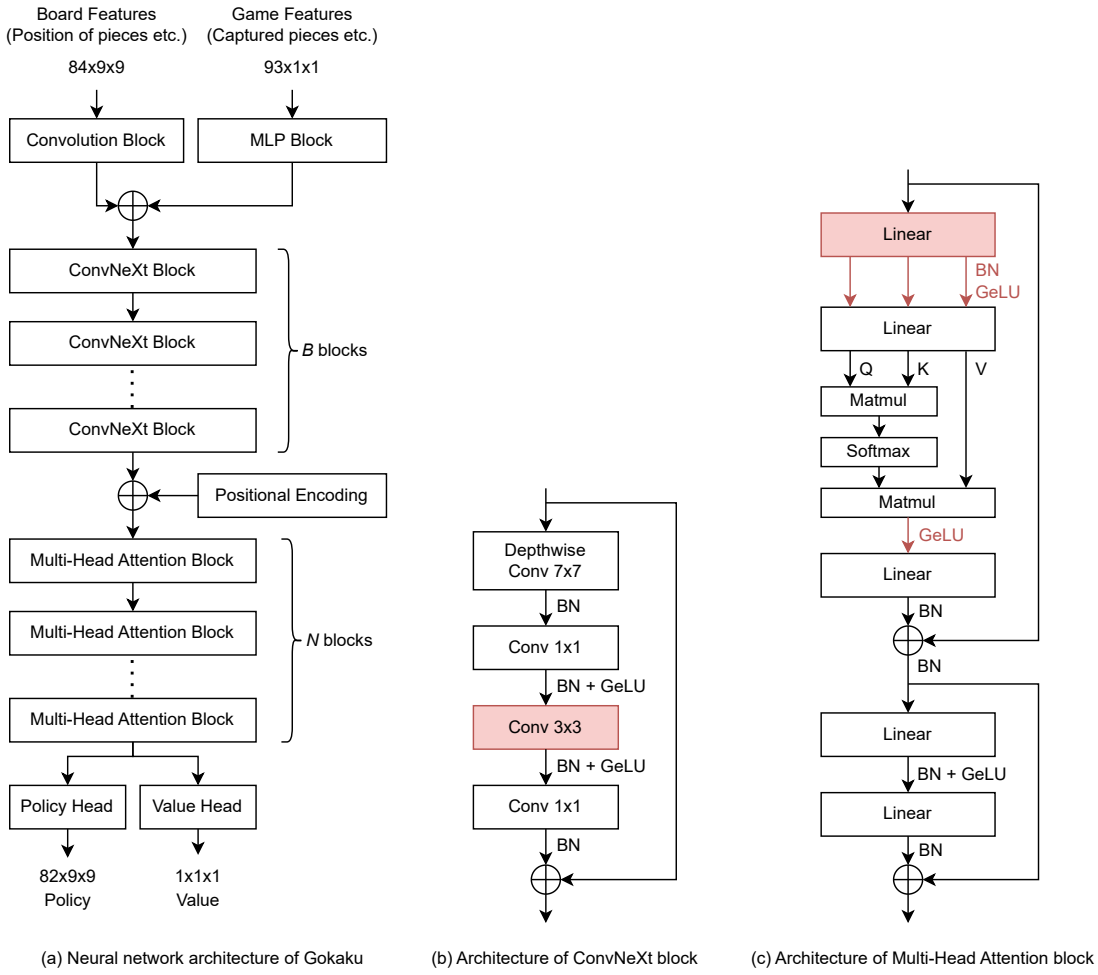


図 1　盤面評価のためのニューラルネットワークの構造（赤色の部分は既存手法からの変更箇所）

Gokaku の探索手法や強化学習手法は既存手法を一部修正したものであるが、盤面評価に用いるニューラルネットワークには前例のない独自の構造を採用している。そこで、ここでは Gokaku で採用しているニューラルネットワーク構造の詳細について述べる。また、このニューラルネットワークを用いた盤面解析の例を紹介する。

2　Gokaku で用いるニューラルネットワークの構造

図 1 に Gokaku で用いるニューラルネットワークの概要を示す。まず、将棋盤面の特徴量を畳み込み層によって変換し、これに持ち駒の特徴量を全結合層で変換した結果を加算して ConvNeXt ブロックへ入力する。次に、ConvNeXt ブロックで得られた特徴量に盤面位置の情報（Positional Encoding を 2 次元に拡張した値）を加算し、Multi-Head Attention ブロックへ入力する。Multi-Head Attention ブロックによって生成された特徴量は、最終的に Policy Head および Value Head に入力さ

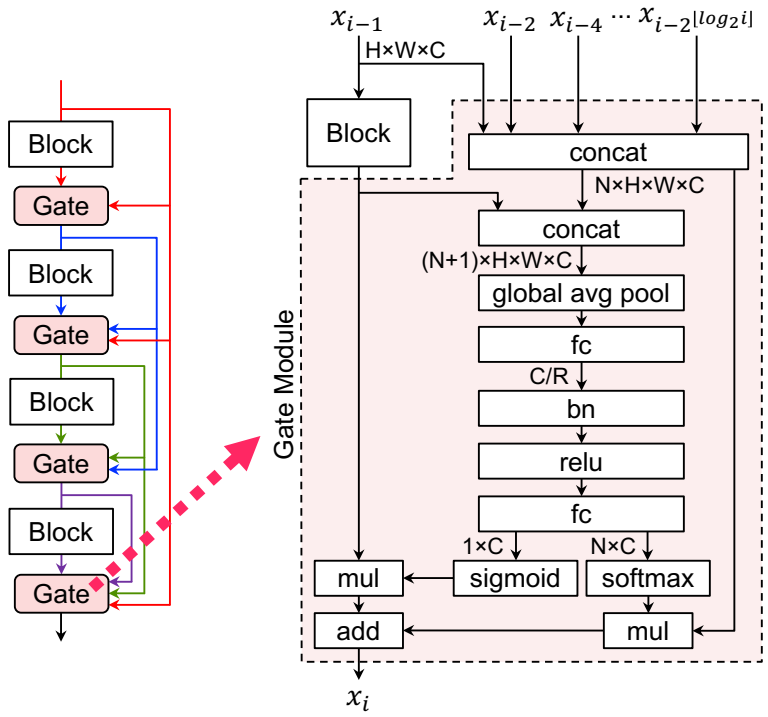


図 2　Skip Connetion 構造

れる。ConvNeXt ブロックと Multi-Head Attention ブロックはそれぞれ Attention 構造をもった Skip Connection 構造 [3] となっており、通常の Residual Connection よりも高い表現力を有することで盤面評価の予測精度を改善している。

ConvNeXt ブロックには、7 × 7 の Depthwise Convolution 層と 3 × 3 の Convolution 層が含まれる。7 × 7 の Depthwise Convolution 層は広域の特徴量を集約し、3 × 3 の Convolution 層は局所的な特徴量の統合を担う。理論的には、3 つの ConvNeXt ブロックを用いることで盤面全体の情報を捉えることが可能である。ConvNeXt ブロックは既存手法 [1] を参考に設計した構造であるが、3層目のConvolution層のカーネルサイズが3×3である点、及び、すべてのConvolution層のチャンネル数が同じである点が異なる。これは、コンピュータ囲碁のプログラムである「Maru」において実験評価を行い、その結果から最も棋力の改善が見られた構造を採用したものである。

ConvNeXt ブロックの出力に対して位置情報を加算したものを、Multi-Head Attention ブロックの入力とする。ここでの位置情報とは、Transformer 系のニューラルネットワークで用いられる Positional Encoding を 2 次元に拡張したものであり、縦方向の位置情報と横方向の位置情報をそれぞれ独立に計算し、これを加算することで盤面上の位置情報を表現している。また、Multi-Head Attention ブロックの構造も一般的な Transformer 系のアーキテクチャ [2] と異なり、Query、Key、Valueの生成処理と統合処理に活性化関数を導入することで非線形変換を行うように工夫している。この修正により、必要となる計算量は増加するが、ニューラルネットワークの表現力が改善するため、ニューラルネットワークによる盤面評価の精度向上と棋力の改善が期待できる。

ConvNeXt ブロックと Multi-Head Attention ブロックは独自に開発したSkip Connection構造で接続

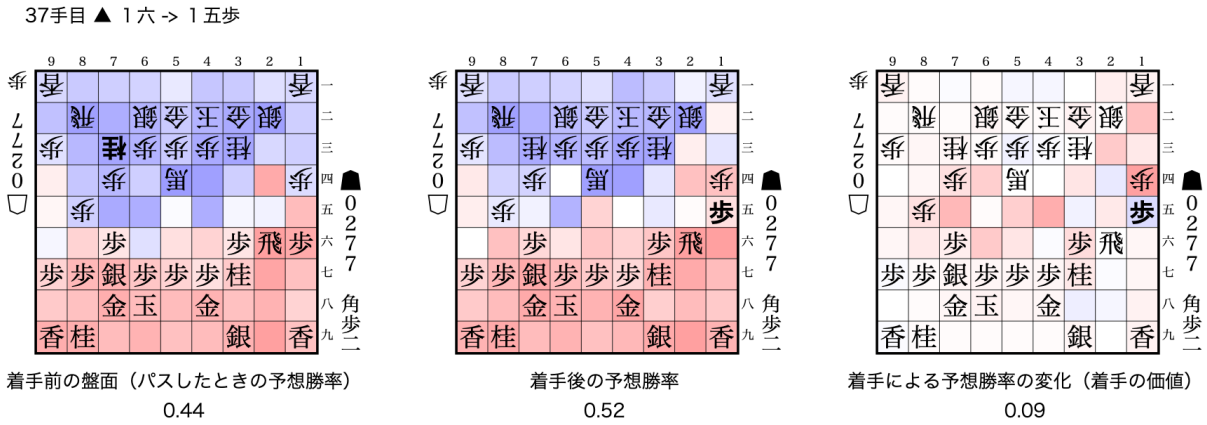


図 3　盤面解析の例（「１五歩」による評価値の変化）

されている。図 2 に Skip Connection 構造の概要を示す。Skip Connection 構造では、それぞれのブロックは 2^n（n は 0 以上の整数）個だけ前のブロックの出力を受け取り、これらの受け取った特徴量からそのブロックの出力に加算すべき特徴量を Gate Module を用いて抽出する仕組みとなっている。Gate Module は Attention 構造となっており、特徴量の抽出を行うが、伝播信号全体の減衰や増幅が発生しないように設計されている。そのため、この Gate Module を導入したとしても勾配消失や勾配爆発は発生せず、通常の Residual Connection と同様に安定した学習が可能である（詳細は Appendix A を参照）。また、Skip Connection構造は通常のResidual Connectionよりも表現力が高く、より複雑な盤面の状態を表現できるため、盤面評価の精度向上が期待できる。

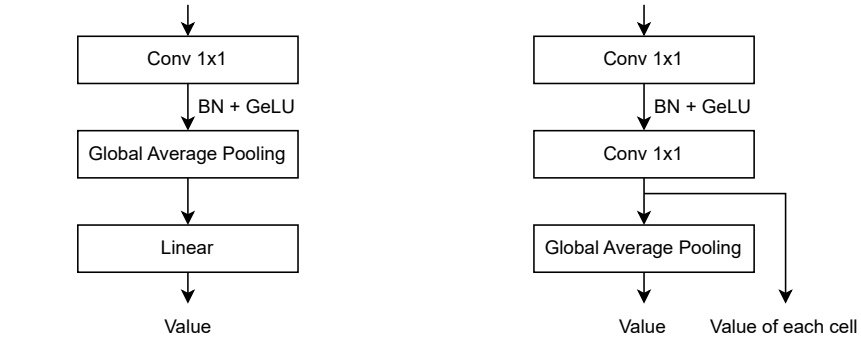
Multi-Head Attention ブロックから得られた特徴量は、Policy Head と Value Head の 2 つに分岐して入力される。Policy Head は、移動する駒の種類・移動方向・移動先座標ごとの確率（$82 \times 9 \times 9$）を出力する。また、Value Head ではその盤面における予測勝率を出力する。これは、深層学習を用いたゲーム人工知能において一般的に採用されている指標と同様であり、探索の優先度付けや最終的な着手決定の基準となる。

3　Gokaku の盤面解析

第 35 回世界コンピュータ将棋選手権の開催期間中（2025 年 5 月 3 日〜5 日）は、ここで述べる盤面解析機能のデモを `https://takeda-lab.jp/ga/` にて公開している。

Gokaku のニューラルネットワークの Value Head は、全体の予測勝率を出力するだけではなく、盤面のそれぞれの座標についての予測勝率も出力する構造になっている。これは、Class Activation Mapping (CAM) [7] を参考にして設計したものであり、それぞれの座標についての予測勝率を可視化することにより、盤面での駒の働きや、局面の形勢を視覚的に理解することができる。図 3 に盤面解析の様子を示す。この図は、Gokaku のニューラルネットワークの Value Head から出力された盤面のそれぞれの座標についての予測勝率を可視化したものであり、赤色は先手番の予測勝率が高い座標を示し、青色は後手番の予測勝率が高い座標を示している。

図 3 左図は着手せずに相手に手番を渡した場合の予測勝率を示しており、図 3 中図は「１五歩」を



(a) Normal Architecture for classification tasks　　(b) Value Head Architecture in Gokaku

図 4　Gokaku の Value Head ブロックの構造

指した場合の予測勝率を示している。着手しなかった場合の予測勝率と着手した場合の予測勝率を比較することにより、その着手が局面に与える影響を視覚的に理解することができる。図 3 右図は「１五歩」を指したことによる予測勝率の変化を示している。この「１五歩」は端攻めを意図した着手であり、１五地点の予測勝率は低下しているが、１二地点や１四地点の予測勝率は上昇しており、これらを起点とした端攻めを意図した着手であることがうかがえる。また、この「１五歩」には「同歩」と応じることを予想しており、手番を握ることにより、６筋や７筋の歩突きも視野に入れていることがわかる。このように、Gokaku の Value Head から出力される座標ごとの予測勝率を可視化することにより、Gokakuの予測勝率の根拠を視覚的に解析することができる。

Gokaku では、Value Head に CAM を参考にした構造を採用することにより、盤面のそれぞれの座標についての予測勝率を出力することができる。図 4 に Gokaku の Value Head の構造を示す。Gokaku の Value Head は、盤面を評価して予測勝率を出力するものであり、その計算手順は分類タスクや回帰タスクのためのニューラルネットワークと同様である。画像分類などのタスクで用いられるニューラルネットワークの Head では、それまでに計算した特徴量を Global Average Pooling 層で集約し、これを全結合層で必要な出力数に変換したものを Sigmoid や Softmax などの正規化関数に入力することで最終的な出力を得る。すなわち、Head の出力を $\mathbf{V}$ とすると、画像分類などのタスクで用いられるニューラルネットワークでは

$$\mathbf{V} = \phi(\mathbf{W_2}\mathrm{GAP}(\sigma(\mathbf{W_1}\mathbf{X} + \mathbf{b_1})) + \mathbf{b_2}), \tag{1}$$

となる。ここで、σ は活性化関数、ϕ は正規化関数、$\mathbf{W_1}$ と $\mathbf{b_1}$ は第 1 層の重みとバイアス、$\mathbf{W_2}$ と $\mathbf{b_2}$ は第 2 層の重みとバイアス、GAP は Global Average Pooling 層の計算を表す。

一方、Gokaku の Value Head では、それまでに計算した特徴量を 1×1 の畳み込み層で必要な出力数に変換し、これを Global Average Pooling 層で集約したものを Sigmoid や Softmax などの正規化関数に入力することで最終的な出力を得る。すなわち、Gokaku の Value Head では

$$\mathbf{V} = \phi(\mathrm{GAP}(\mathbf{W_2}\sigma(\mathbf{W_1}\mathbf{X} + \mathbf{b_1}) + \mathbf{b_2})), \tag{2}$$

となる。式 1、2 ともに、活性化関数 σ と正規化関数 ϕ 以外は線形変換であり、第 2 層の計算と Global

Average Pooling 層の計算順序は交換可能である。すなわち、式 1 と式 2 の計算結果は同じであり、Gokaku の Value Head は画像分類のニューラルネットワークの Head と同じ計算を実行する。

Gokaku では、学習や推論時には Global Average Pooling 層で集約した予測勝率を用いるが、盤面解析時には Global Average Pooling 層で集約する前の特徴量から計算した座標ごとの予測勝率を出力する。Head に入力される特徴量には盤面のそれぞれの座標についての情報が集約されているため、それぞれの座標についての情報を Value Head で予測勝率に変換することにより、それぞれの座標の予測勝率を計算できると考えられる。

4 Gokaku の棋力

開発中のプログラムに大きな変更を加え、2025 年 4 月 2 日より新しい強化学習フェーズを開始した。この強化学習では、5 個の GPU（RTX3090：3 個、RTX A5000：2 個）を用いて自己対戦による棋譜生成と生成した棋譜によるニューラルネットワークの学習を行っている。第 35 回世界コンピュータ将棋選手権までの約 1 ヶ月の学習期間となるため、十分な棋力向上は見込めないが、選手権までにプロ棋士と同等の棋力を持つプログラムの実現を目指して開発を進めている。

参考文献

[1] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, Vol. 30, , 2017.

[3] 武田敦志. Gate module を導入した画像認識のための residual neural network の提案と評価. 第 25 回 画像の認識・理解シンポジウム (MIRU2022), 2022.

[4] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, Vol. 550, No. 7676, pp. 354–359, 2017.

[5] David J Wu. Accelerating self-play learning in go. *arXiv preprint arXiv:1902.10565*, 2019.

[6] Ivo Danihelka, Arthur Guez, Julian Schrittwieser, and David Silver. Policy improvement by planning with gumbel. In *International Conference on Learning Representations*, 2022.

[7] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929, 2016.

A　Skip Connection 構造の学習安定性について

Skip Connection 構造のニューラルネットワークでは、それぞれのブロックは 2^n（n は 0 以上の整数）個だけ前のブロックの出力を受け取り、これらの特徴量からそのブロックの出力に加算すべき特徴量を Gate Module を用いて抽出する仕組みとなっている。i 番目のブロックの出力を $\mathbf{x}_i$ とすると、

$$
\begin{aligned}
\mathbf{y}_i &= f_i(\mathbf{x}_{i-1}; \boldsymbol{\theta}) \\
\mathbf{p}_i &= (\mathbf{y}_i, \mathbf{x}_{i-1}, \mathbf{x}_{i-2}, \mathbf{x}_{i-4}, \cdots, \mathbf{x}_{i-2^{N-1}}) \\
\mathbf{q}_i &= (\mathbf{x}_{i-1}, \mathbf{x}_{i-2}, \mathbf{x}_{i-4}, \cdots, \mathbf{x}_{i-2^{N-1}}) \\
\mathbf{s}_i &= \mathit{sigmoid}(g_i(\mathit{global\text{-}pooling}(\mathbf{p}_i); \boldsymbol{\theta})) \\
\mathbf{w}_i &= \mathit{softmax}(h_i(\mathit{global\text{-}pooling}(\mathbf{p}_i); \boldsymbol{\theta})) \\
\mathbf{x}_i &= \mathbf{s}_i \circ \mathbf{y}_i + \mathbf{w}_i \cdot \mathbf{q}_i
\end{aligned}
$$

となる。ここで、N は入力値の個数、$\boldsymbol{\theta}$ はニューラルネットワークの学習パラメータであり、関数 $f(\cdot)$・$g(\cdot)$・$h(\cdot)$ は畳み込み層などを用いたブロック内の計算を表す。ここで、$\mathbf{w}_i = (\mathbf{v}_{i,0}, \mathbf{v}_{i,1}, \cdots, \mathbf{v}_{i,N-1})$ とすると、i 番目のブロックへの入力値の個数が $\lfloor \log_2 i \rfloor$ 個であるため、i 番目のブロックの出力値 $\mathbf{x}_i$ は

$$
\mathbf{x}_i = \mathbf{s}_i \circ f_i(\mathbf{x}_{i-1}; \theta) + \sum_{j=0}^{\lfloor \log_2 i \rfloor} \mathbf{v}_{i,j} \circ \mathbf{x}_{i-2^j} \tag{3}
$$

となる。ここで、$\sum_j \mathbf{v}_{i,j} = \mathbf{1}$（$\mathbf{1}$ はすべての要素が 1 の行列）であることに着目して式 3 を変形すると、i 番目の残差ブロックの出力値は

$$
\mathbf{x}_i = \mathcal{F}_i(\mathbf{x}_0; \boldsymbol{\theta}) + \mathbf{x}_0 \tag{4}
$$

となる。ここで、関数 $\mathcal{F}_i(\cdot)$ は 1 番目から i 番目までのブロックで実行される学習パラメータに依存した計算を表す。Skip Connection 構造のニューラルネットワークでは、1 個目のブロックに入力された特徴量 $\mathbf{x}_0$ がすべてのブロックに直接伝播されるため、入力特徴量の特性を保持したまま各ブロックでの計算を行うことができる。また、学習時においても、最後のブロックに入力された勾配情報がすべてのブロックに直接伝播されるため、勾配消失問題や勾配爆発問題が発生せず、安定した学習が可能である。なお、数学的帰納法を用いると式 3 が式 4 に変形可能であることを証明できる。

余談

$\sum_j \mathbf{v}_{i,j} = \mathbf{1}$ が学習安定の条件であり、この条件を満たすニューラルネットワークは容易に多層化できる。例えば、Pre-Activation 以降の ResNet の場合、

$$
\mathbf{v}_{i,j} = \begin{cases} \mathbf{1} & (j = 0) \\ \mathbf{0} & (j \neq 0) \end{cases}
$$

であるため、$\sum_j \mathbf{v}_{i,j} = 1$ が成り立つ。一方、初期の Transformer [2] では、加算後に Layer Normalization を行っており、式 4 が成立しないため、多層化が困難であったと考えられる。