FPGAベースのCNN推論アクセラレータの一検討

印藤 俊介[1,a)]　中原 康宏[1,b)]　境 琳太郎[1,c)]　飯田 全広[1,d)]

概要：AI エッジコンピューティングにおいて，FPGA は高エネルギー効率，高並列性や回路の再構成が可能などの点から，CPU や GPU プラットフォームと比較して優位性を示している．しかし年々発展する CNN モデルに伴いその演算コストとパラメータサイズは膨大になっているため，単一 FPGA ではなく複数の FPGA を利用することで大規模な CNN モデルを効率的に処理することが考えられる．本論文では FPGA クラスタの実装に向けた一検討として，単体 FPGA で DRAM データ転送の測定と CNN 推論処理性能の評価を行う．結果として VGG16 を対象としたとき，畳み込み層では処理速度が向上したが，全結合層を含めた全体の処理では速度の低下が見られた．しかしながら，DRAM データ転送の改善や CNN モデルの選択により，さらなる性能の改善が見込まれる．

キーワード：特定用途向けアーキテクチャ，リコンフィギュアラブルコンピューティング，FPGA

A Study of FPGA-based CNN Inference Accelerator

Shusnuke Into[1,a)]　Yasuhiro Nakahara[1,b)]　Rintaro Sakai[1,c)]　Masahiro IIDA[1,d)]

Abstract: In AI edge computing, FPGAs have shown advantages over CPU and GPU platforms in terms of high energy efficiency, high parallelism, and circuit reconfigurability. However, as the computational cost and parameter size of CNN models are increasing year by year, multiple FPGA system instead of a single FPGA can be used to efficiently process large-scale CNN models. In this paper, we measure the DRAM data transfer and evaluated the CNN inference processing performance on a single FPGA as a study for implementing FPGA clusters. The results showed that the processing speed of the convolutional layer improved on the VGG16, but the processing speed of all layers decreased. However, further performance improvement is expected by improving DRAM data transfer and selecting CNN models.

Keywords: Application-specific architecture, Reconfigurable computing, FPGA

1. はじめに

CNN（Convolutional Neural Networks）は，主にコンピュータビジョンや音声処理の分野で利用されており，IoT（Internet of Things）によって増加するセンサ等から発生するデータを処理するアプリケーションとして有望視されている．また，ますます増加していくデータのにおいて，処理遅延を削減したりデータの通信量を抑えるために，CNN の推論処理を端末側で処理する AI エッジコンピューティングが注目を浴びている．エッジコンピューティングにおいて，FPGA（Field-Programmable Gate Array）は高エネルギー効率，高並列性や回路の再構成が可能などの点から，CPU（Central Processing Unit）や GPU（Graphics Processing Unit）プラットフォームと比較して優位性を示している [1][2]．しかし年々発展する CNN モデルに伴いその演算コストとパラメータサイズは膨大になっており，単一 FPGA では効率的に処理することが困難になっている．そこで複数の FPGA を互いに接続しクラスタ化することで，大規模な CNN モデルを効率的に処理することが考えられる．

本論文では FPGA クラスタの実装に向けた一検討として，単体 FPGA で CNN 推論処理性能の評価を行う．CNN 処理を行うためのアーキテクチャとして，AI チップ ReNA[7]

[1] 熊本大学, Chuo-ku, Kumamoto-shi 860–8555, Japan
[a)] into@st.cs.kumamto-u.ac.jp
[b)] nakahara@st.cs.kumamoto-u.ac.jp
[c)] c7811@st.cs.kumamoto-u.ac.jp
[d)] iida@cs.kumamoto-u.ac.jp

© 2021 Information Processing Society of Japan

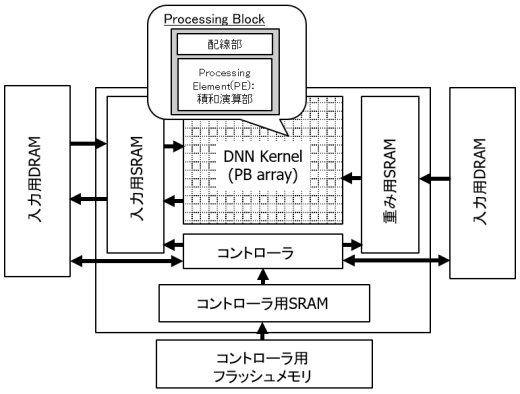

図 1 ReNA の概要図

のアーキテクチャを FPGA に合わせて実装する．ReNA は畳み込み層と全結合層を同一の構造で効率的に処理することが可能な CNN 推論用アクセラレータである．本システムは主に Verilog HDL を用いて RTL ベースの実装を行う．評価として，実機でのデータ転送性能の測定とシミュレータを用いた CNN 推論処理速度の見積りを行う．

2. 関連研究

大規模 CNN を処理するとき，データは外部メモリから転送する必要があるため，外部メモリとのデータ転送を減らすことは，処理速度と電力消費の点でアクセラレータの性能を大幅に向上させることができる。

CNN を効率的に処理するための手法の一つは，モデルのサイズを小さくすることである．Zhao らは，重みとアクティベーションを 2 値まで量子化する 2 値化ニューラルネットワークを FPGA に実装することで，メモリ要件を大幅に削減した [3]． Lu らは，モデルにネットワークの精度に大きく寄与しないニューロンを排除するプルーニングを行ったニューラルネットワークを使用することで，高速かつ効率的な推論を実行した [4].

また別の手法として，オンチップでデータを再利用することで外部メモリとのデータ転送を削減する研究も多くある．Aydonat らは，CNN 推論をシストリックアレイアーキテクチャにマッピングすることでデータの再利用性を高めている [5]．Wei らは，オンチップメモリの利用効率を上げる LCMM フレームワークを提案し，データの再利用によりオンチップの利用率を高め，オンチップメモリの容量を最小化する [6].

これらの研究は単体の FPGA に対して最適化されている．本研究では，複数の FPGA を高速インタフェースにより接続することで FPGA クラスタを構築し，大量のリソースを用いて大規模 CNN 推論処理を効率的に実行することを目的とする．本論文ではその検討として FPGA 当たりの CNN 推論性能を評価する．

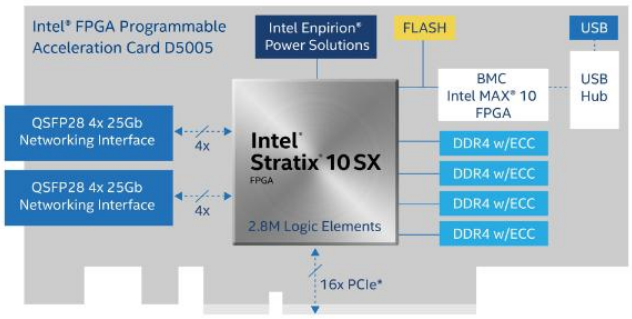

図 2 FPGA ボード

表 1 Intel FPGA PAC D5005 の仕様

FPGA	Intel Stratix 10 SX FPGA
オフチップメモリ	8 GB DDR4-2400 ×4
インタフェース	PCIe Gen3×16
	QSFP28 ×2

表 2 Intel Stratix 10 SX FPGA の仕様

プロセステクノロジ	14 nm
LE	2,800 K LEs
DSP	5,760 個
オンチップメモリ	244 Mb

3. FPGA 実装

3.1 ReNA

ReNA の概要図を図 1 に示す．ReNA は ASIC 向けに設計された CNN 推論用アクセラレータである．回路の再構成により，畳み込み層と全結合層を同一の構造で効率的に処理することが可能な設計となっている．図 1 にあるように，ReNA は入力，重み，コントローラ制御用の 3 つの SRAM を持つ．計算は PB array で行われる．PB array は処理要素である PB（Processing Block）が 64x64 個アレイ状に並べられており，それぞれの PB が一度に一つの積和演算を行うことができる．PB は，入出力を制御するためのモジュールと，実際に積和演算を行う PE（Processing Element）により，構成され，入出力制御モジュールにより PB array の接続を切りかえ，回路を再構成することができる．PE での計算は 8 bit 固定小数点で行われる．

4. 提案システム

本研究で使用される FPGA ボード（図 2）は，Intel FPGA PAC D5005 である．このボードはデータセンタ向けの PCI Express ベースのアクセラレーション・カードである．表 1 に FPGA ボードの仕様を示す．FPGA として Intel Stratix 10 SX FPGA（表 2）を備える．Stratix 10 SX FPGA は 5,760 個の DSP ブロック（Digital Signal Processing Block ）を持つ．DSP ブロックは FPGA 上に固定機能として持つ演算器で，通常ロジックを構成する LUT（Look-up Table）を用

© 2021 Information Processing Society of Japan

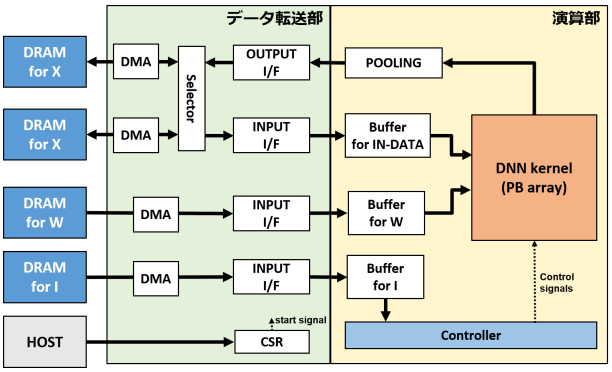

図 3　システムの概要図

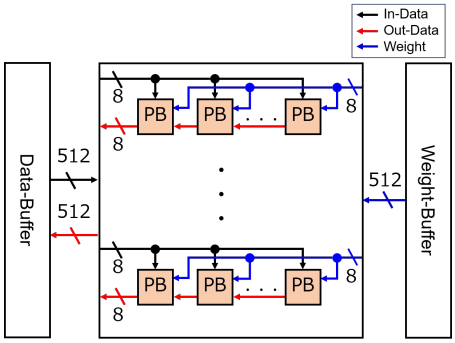

図 4　FPGA 実装での PB array

いるよりも，高速かつ省消費電力で動作させることができる．またこの FPGA の DSP ブロックは乗算器のサイズが 18x19 以下で実装する場合，1/2 の DSP ブロックリソースで実装することができる．そのため，その場合実質的に倍の 11,520 個の DSP ブロック利用することが可能となる．今回は，CNN の大量の積和演算をこの DSP ブロックにより処理することで，より高性能に演算を行う．また，外部メモリは，8GB の DDR4 メモリが 4 枚あり，1 回の要求で 512bit データを 64 回バースト転送することが可能となっている．

4.1　FPGA ボード

4.2　システム構成

設計したシステムの概要図を図 3 に示す．図中 DRAM の X，W，I の記号はそれぞれ，入出力データ（X），重みデータ（W），命令 (I) を表している．構成としては，DRAM との制御を行うデータ転送部と CNN 処理を行う演算部に分けられる．データ転送部と演算部はそれぞれ独立して動作しており，PB array で演算などが処理されているとき，データ転送部は常に DRAM とのデータ通信を行う．そうすることで，DRAM とのデータ通信による遅延を抑える．

次に演算部について説明する．演算部の制御はコントローラによって行われる．コントローラは命令バッファからマイクロコードを読み出し，それに従い制御信号を生成してそれぞれやバッファや PB array の制御を行う．命令は毎クロック呼び出すのではなく，一つの命令で一連の処理

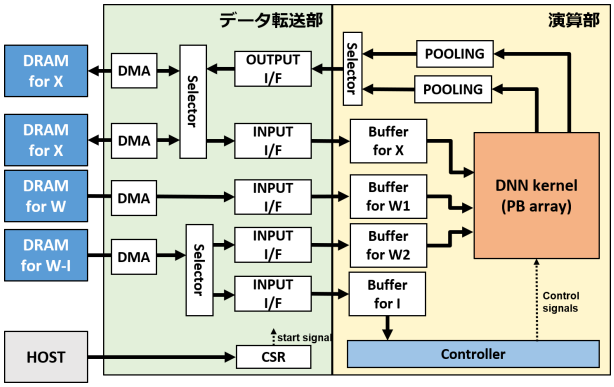

図 5　PB あたりの乗算器を 2 個にしたシステムの概要図

を行い，その処理が終了したら次のマイクロコードを読みだす．そのため，命令は入力データや重みデータと比較して DRAM からのデータ転送は少ない．

次に，FPGA 実装での PB array について説明する．PB array の構造を図 4 に示す．PB array は元の ReNA 同様，64 x 64 の PB で構成され，各 PB 内の演算も 8bit の固定小数点データを用いて処理される．入力，重み，出力データの配線幅はそれぞれ外部から 512bit で入出力され，各行に 8bit ずつ 64 分割されて接続される．各 PB には FPGA に備わっている DSP ブロックが割り当てられ，合計 4,096 個の PB が利用される．

また，今回使用する FPGA ボードでは 4.1 節で説明しているように，1/2 の DSP ブロックリソースで乗算器を実装できるため，各 PB に 2 つの乗算器を割り当てた PB array の設計も行う．2 つの乗算器を割り当てる場合，同じ PB 内での演算は同じ入力データに対して異なる重みデータを乗算して処理される．そのため，重みデータは元の 2 倍必要になる．重みデータの増加によるデータ転送のロスを減らすため図 5 のようにシステムの構成も変更する．この実装では 2 倍必要になる重みデータに対応させるため，もともと命令に割り当てていた DRAM の中にも重みデータを割り当てる．DRAM から転送されたデータは内部のセレクタによって重みバッファか命令バッファかを選択する．命令はもともとデータの転送量が少ないため，共有することによる影響は少ない．この変更により演算の並列度は 2 倍の 8,192 に増加するが，配線の増加により動作周波数は減少が予想される．

4.3　処理フロー

ReNA が畳み込み層と全結合層を PB array でどのように処理するか説明する．

4.3.1　畳み込み層

畳み込み層の処理の流れを図 6 に示す．CNN の処理は図 6（a），（b），（c）の順で行われる．まず（a）でそれぞれの PB に対して入力データを転送し内部レジスタに保存する．次に（b）で重みを転送しそれと保存されている入

© 2021 Information Processing Society of Japan

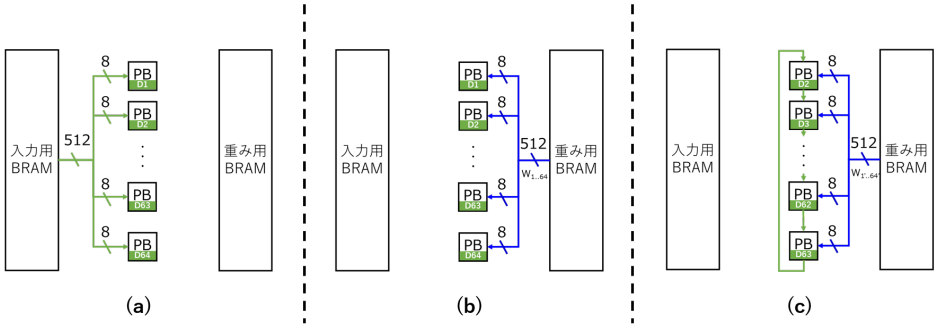

図 6　畳み込み層処理の流れ

力データで積和演算を行う．畳み込み層での演算の時は，アレイの行ごとに同じ重みを利用するため 1 回の計算に対して重みの転送は 1 度行えばよい．その後（c）で各 PB が保持している入力データを一つ下の PB に転送し，転送された新たな重みと積和演算を行う．この処理を繰り返すことで，各 PB に出力データが求められる．求められた出力データは，左の PB にバケツリレー方式でデータを転送しながら取り出される．つまりこの処理では，最初に 64 サイクルかけてそれぞれの PB に入力データを転送し，データを移動させながら積和演算を行う処理を 64 サイクル繰り返すことで結果を求められ，64 サイクルかけて出力データの転送が行われる．そのため，一連の計算が終了するまで，64x3 サイクルと制御のための数サイクルが必要になる．

以上の処理を順次実行すると，入出力データの転送中は PB の積和演算が行われず演算器が動作しない時間が生じる．それを解決するために，ReNA では入力データの転送，積和演算，出力データの転送を独立して行うことができるようになっており，それによりデータの転送を隠蔽することができる．このようにすることで，最初を除いて常に演算が行われ効率的に処理することが可能となる．

4.3.2　全結合層

全結合層の処理の流れを図 7 に示す．ここでは説明のため PB array を 3x3 としている．畳み込み層では縦の 1 列でデータを移動させながら処理を行っていたが，全結合層ではアレイ全体でデータを移動させながら処理を行う．まず演算を行う前に，（a）で各 PB が次にデータを送る先の PB を設定するために，転送先設定用データをバケツリレー方式で転送する．このデータは重みのメモリの中に保存されている．その結果，図 7 のように，横方向も含めた一つの輪が形成される．後は畳み込み層と同じように，全ての PB に対して入力データを転送した後，重みデータを転送して積和演算，次の PB にデータを転送という処理を繰り返す．ここで，積和演算の際に畳み込み層では 1 度の演算につき重みの転送は 1 回で済んでいたが，全結合層処理の場合は演算に用いる重みの値が全て異なるため，入力データと同じように 1 列ずつにデータを送る必要がある．つまり，図の 3x3 の例では 3 サイクルかけて重みを全ての PB に転送

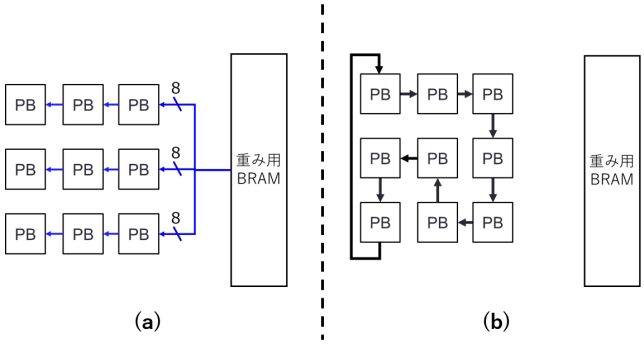

図 7　全結合層処理の流れ

したのち，積和演算が行われる．そして輪になっている PB の数だけ演算とデータの移動を繰り返したとき，各 PB に演算結果が出力される．その後，畳み込み層と同様バケツリレー方式でデータを取り出すことで計算が終了する．

5.　評価

5.1　評価項目

評価では，DRAM データ転送サイクルの測定と，CNN 推論処理速度の見積りを行う．DRAM データ転送のサイクル数は，デザイン内に組み込まれたカウンタ回路によって計測される．計測は ReNA から DRAM に対して 512bit を 64 バーストで転送要求するように信号が送られてから，転送終了信号を受け取るまでを測定範囲とし，50 回行ってその平均を算出する．CNN 処理の見積りには ReNA と同時に開発されたシミュレータを用いる．このシミュレータは ONNX ファイルからモデルのパラメータと重みを取得し，性能を推定する．今回の見積りでは，ImageNet[8] を VGG16[9] で実行したときの処理速度を算出する．VGG16 は畳み込み層 13 層，全結合層 3 層で構成されるモデルである．測定には各 PB に乗算器が 1 個ずつのシステムと 2 個に増やしたシステムについて行い，CNN 推論処理速度は畳み込み層のみを処理した場合と，全結合層を含む全層を処理した場合に分けて見積りを行う．

5.2　評価結果

表 3 にリソース使用率を示し，表 4 に VGG 処理速度の結果を示す．まず，DRAM データ転送について，DRAM

© 2021 Information Processing Society of Japan

表 3　リソース使用率

	BRAM [Mb]（使用率%）	DSP	Used logics（使用率%）
FPGA ReNA	9.99 (4%)	4,096	303,615 (33%)
FPGA ReNA 2-mult	9.92 (4%)	4,096	319,426 (34%)

転送に消費される平均転送サイクル数は 205 となった．これは，FPGA これは乗算器を 2 倍に増やした構成でも同様の通信を行っているため共通である．この結果から，PB array での演算がおよそ 64 サイクル必要であるのに対して，約 3 倍のサイクルを必要とすることがわかる．そのため，データの隠蔽の効果が DRAM のデータ転送によって大幅に影響を受ける．

次に CNN 推論処理速度について，まず各 PB に 1 つの乗算器を備えたシステムでは，元の ASIC ReNA と比較して動作周波数は半分以下の 144MHz となった．また FPS は，畳み込み層では 1.6 向上したが，全層の処理では 0.4 の低下がみられた．今回 ASIC から FPGA に変更するにあたり，演算部においてバッファから PB array へのデータ転送を 2bit データを 4 回から 8bit データを 1 回にすることで転送速度を 4 倍にしたにもかかわらず，全層の処理速度は低下した．この原因は DRAM データ転送によるロスだと考えられる．先に説明したように，データ転送の隠蔽の効果が小さく，畳み込み層ではそれを踏まえても性能の向上がみられるが，全層では全結合層の演算に必要な重みデータ量が大きいことが影響で速度が低下した．次に乗算器を 2 倍にしたシステムでは，乗算器 1 個のシステムと比べ，動作周波数がおよそ 66%の 95MHz となった．FPS は畳み込み層ではさらなる向上が見られたが，全層では動作周波数の減少と全結合層で乗算器を 2 倍にする恩恵が受けられないことが影響し，さらに低下する結果となった．

これらの結果から DRAM データ転送が CNN 処理速度の向上を大きく妨げている．そのため，データ転送部の周波数を変更するなどして，データ転送のロスを減らすことで処理速度が向上する見込みがある．また，利用するモデルの選択によっても利点が生まれる可能性がある．今回見積りに利用した vgg16 は全 16 層の内，全結合層を 3 層含んでいるが，モデルによっては全結合層を含まないモデルもある．例えば最終出力を画像とするモデルであれば，出力層として全結合層は必要ない．画像の認識などの用途でも最後に全結合層を含まない手法も存在する [10]．そのようなモデルを利用することで優位性のあるアプリケーションを対象とすることで，より FPGA に適した処理を行う．

6. おわりに

本論文では FPGA クラスタの実装に向けた一検討として，単体 FPGA に AI チップ ReAN のアーキテクチャを FPGA に合わせて実装し，DRAM データ転送の測定と CNN 推論処理性能の評価を行った．結果として VGG16 を対象としたとき，畳み込み層では処理速度が向上したが，全結合層を含めた全体の処理では速度の低下が見られた．しかしながら，DRAM データ転送の改善や CNN モデルの選択により，さらなる性能の改善が見込まれる．

表 4　VGG 処理速度

	FPGA ReNA	FPGA ReNA 2-mult	ASIC ReNA
CNN モデル	VGG16		
データセット	ImageNet		
演算並列度	4,096	8,192	4,096
DRAM 転送サイクル	205 Cycle		-
動作周波数	144 MHz	95 MHz	350 MHz
FPS 畳み込み層	10.5	12.2	8.9
FPS 全層	7.0	6.5	7.4

参考文献

[1] Biookaghazadeh, Saman, Ming Zhao, and Fengbo Ren. ”Are fpgas suitable for edge computing?.” Workshop on Hot Topics in Edge Computing (HotEdge 18). 2018.

[2] Al-Ali, Firas, et al. ”Novel Casestudy and Benchmarking of AlexNet for Edge AI: From CPU and GPU to FPGA.” IEEE Canadian Conference on Electrical and Computer Engineering (CCECE). IEEE. 2020

[3] Ritchie Zhao, et al. ”Accelerating binarized convolutional neural networks with software-programmable fpgas.” Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. 2017.

[4] Liqiang Lu, et al. ”An efficient hardware accelerator for sparse convolutional neural networks on fpgas.” 2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE, 2019.

[5] Utku Aydonat, et al. ”An opencl™ deep learning accelerator on arria 10.” Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. 2017.

[6] Xuechao Wei, Yun Liang, and Jason Cong. ”Overcoming data transfer bottlenecks in FPGA-based DNN accelerators via layer conscious memory management.” 2019 56th ACM/IEEE Design Automation Conference (DAC). IEEE, 2019.

[7] 中原康宏，千竈順太郎，尼崎太樹，趙　謙，飯田全広「AI エッジコンピューティング向け DNN アクセラレータ」信学技報, vol. 119, No. 287, RECONF2019-38, pp. 15-20, 2019

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. FeiFei, “ Imagenet: A large-scale hierarchical image database, ” 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248-255, 2009.

[9] Karen Simonyan, Andrew Zisserman, “ VERY DEEP CONVOLU-TIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNI-TION,”The International Conference on Learning Representations (ICLR), 2015.

[10] M. Lin, Q. Chen, and S. Yan. ”Network in network.” In Proc. of ICLR, 2014.

© 2021 Information Processing Society of Japan