Vol.2017-ARC-227 No.15
2017/7/27

高電力効率なCNNアクセラレータ実現に向けたカーネルクラスタリングの応用の検討

進藤 智司[1]　松井 優樹[2]　八巻 隼人[2]　津邑 公暁[1]　三輪 忍[2]

概要：Convolutional Neural Network (CNN) と呼ばれるニューラルネットワークが画像認識の分野で広く利用されている．近年では，認識精度を向上させるために，CNN の規模を増大させる傾向があり，これに伴って，計算時間や消費エネルギーの大きさが問題となっている．そこで，CNN の計算に特化したハードウェアアクセラレータが多数開発されている．その中でも，CNN に含まれる計算を近似することで計算量を削減し，電力効率の向上を図る研究が盛んに行われている．しかし，アプリケーションによっては，CNN の出力精度の低下が無視できない場合も多く，CNN の認識精度の低下を抑えつつ，高い電力効率を持つアクセラレータの実現が望まれている．本稿では，高電力効率な CNN アクセラレータの実現を目指し，CNN に含まれる計算の近似方法を検討する．特に CNN の特徴であるカーネルの畳み込み演算に着目し，カーネルのクラスタリング結果に基づき，クラスタ単位でカーネルの畳込み演算を近似する方法を検討する．VGG や GoogleNet など，現在広く使用されている CNN のカーネルをクラスタリングした結果，異なる層に共通して類似したカーネルが存在することを確認した．このクラスタリング結果に基づき，本稿ではアクセラレータの設計を検討する．

1. はじめに

画像認識において，**Convolutional Neural Network (CNN)** と呼ばれるニューラルネットワークが高い認識精度を示し，広く利用されている．近年では，認識精度を向上させるために，CNN の規模を増大させる傾向があり，これに伴って，計算時間や消費エネルギーの大きさが問題となっている．この問題に対し，CNN の計算に特化したハードウェアアクセラレータが多数開発されている．その中でも，CNN に必要となる多量の積和演算に着目し，積和を並列に計算可能な構造を持つアクセラレータが多く提案されている [1], [2], [3]．これらのアクセラレータは，GPU と比較して約 100 倍の省エネルギーを実現しているが，CNN の大規模化に対応するためには，回路面積を増大させる必要がある．一方，CNN に含まれる計算を近似することで計算量を削減し，単純な回路で構成可能な CNN アクセラレータが提案されている [4]．しかし，アプリケーションによっては，CNN の出力精度の低下が無視できない場合も多く，CNN の認識精度の低下を抑えつつ，高い電力効率を持つアクセラレータの実現が望まれている．

[1] 名古屋工業大学
Nagoya Institute of Technology
[2] 電気通信大学
UEC

本稿では，高電力効率な CNN アクセラレータの実現を目指し，CNN に含まれる計算を近似する方法を検討する．特に CNN の特徴であるカーネルの畳み込み演算に着目する．1 つの CNN 内には数百から数千のカーネルが存在し，カーネルをいくつかのクラスタに分類可能であることが確認されている [5]．本稿では，カーネルクラスタリングの結果に基づき，クラスタ単位でカーネルの畳み込み演算を近似するアクセラレータを検討する．

2. 研究背景

本章では，CNN の概要，および CNN アクセラレータに関する既存研究について述べる．また，CNN の近似方法に関する既存研究として，カーネルクラスタリングについて述べる．

2.1 CNN

ニューラルネットワークとは，生物の脳神経系を模倣した数理モデルであり，**ニューロン**と呼ばれる多数の計算ユニットと**シナプス**と呼ばれるニューロン同士の結合から成る．基本的なニューラルネットワークは層状に並んだニューロンが隣接した層のみと結合した構造を持つ．このニューラルネットワークの構造を**図 1** に示す．

図 1 のネットワークは入力層，中間層，出力層の 3 層

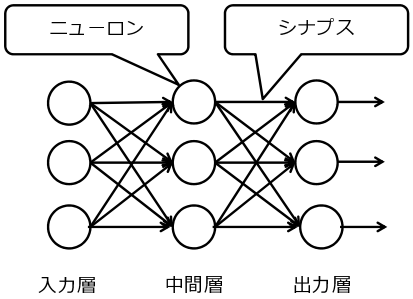


図 1　ニューラルネットワーク

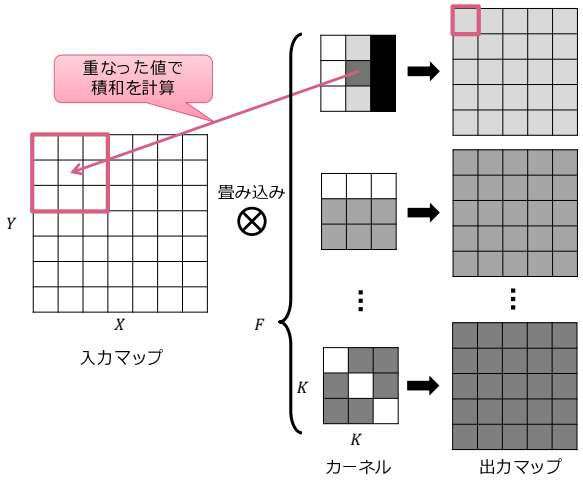


図 2　畳み込み層における計算

からなり，入力層へデータが入力されると入力層に位置するニューロンはシナプスを介して中間層のニューロンへ計算結果を出力し，入力層のニューロンから入力を受け取った中間層のニューロンは出力層のニューロンへ計算結果を出力する．シナプスはそれぞれ**重み**と呼ばれる値 $w_i\ (i = 1, ..., n)$ を持っており，1 ニューロンへの入力を $x_i\ (i = 1, ..., n)$ とすると，出力 y は次の式で計算される．

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{1}$$

ここで，b はバイアスと呼ばれる，ニューロンごとに異なる定数であり，f は活性化関数と呼ばれる，神経細胞の性質をモデル化した関数である．この式からも分かるように，ニューラルネットワークにおける処理は，主に入力値とそれに対応したシナプス重みの積和演算によって構成される．

なお，ニューラルネットワークには様々な構造を持つものが存在しており，そのうちの 1 つである Convolutinal Neural Network (CNN) は，畳み込み層と呼ばれる特別な層を持つニューラルネットワークである．この畳み込み層では，ニューロンの出力を二次元に並べて**マップ**と呼ばれる画像として扱い，重みも同様に**カーネル**と呼ばれる二次元フィルタとして扱う．そして，直前の層から入力されたマップに対してカーネルを畳み込むことでニューロンの出力を求め，それを次の層へまたマップとして渡す．

畳み込み層における計算を**図 2** を用いて説明する．入力マップは横 X 個，縦 Y 個の要素で構成され，これに対して $K \times K$ 個の要素から構成された F 枚のカーネルを畳み込んでいる．畳み込み演算は，入力マップに対してカーネルを重ね，重なった値同士の積を求め，和を計算することで出力マップの 1 要素の値を計算する．これを入力マップ上でカーネルをスライドさせながら行うことで，出力マップの全ての要素の値を計算する．

2.2　CNN アクセラレータ

2.1 節で述べたように，CNN では畳み込み層の計算に多くの積和演算が必要となる．さらに，近年は CNN の認識精度を向上させるために，畳み込み層の数およびカーネルの枚数を増加させる傾向がある．これにより，CNN に必要な計算量が増加してきており，計算に必要な時間および消費電力も増大してきている．これを受けて，CNN の計算に特化したハードウェアアクセラレータが多数開発されている．これらのアクセラレータは主に 2 種類に分類することができる．

1 つ目は，学習済み CNN が持つ構造やパラメタはそのままに，CNN を効率的に計算できるアクセラレータである．例えば，Chen ら [1] は，CNN のシミュレーションに必要な多数の積和演算を並列に計算可能な演算器を備えたコアを設計し，CNN における推論処理を高速に実行するアクセラレータを提案している．さらに，同氏ら [2] はこのコアを複数接続することで，同時に計算可能なニューロンの数を増やし，さらなる高速化を実現している．このタイプのアクセラレータは，学習によって獲得した CNN の構成をそのまま使用するため，その CNN の認識精度は保証される．しかし，計算速度を追求すると，コア数などを増加させる必要があり，回路面積や消費電力が増大してしまう．

2 つ目は，演算量やデータ量が削減されるように CNN の構造やパラメタを変更し，単純な回路で CNN の計算を行うアクセラレータである．近年では，CPU や GPU を用いて CNN をシミュレートする際の，演算量やデータ量を削減することを目指して，CNN が持つパラメタの近似化が盛んに研究されている．CNN が持つパラメタの近似方法の 1 つに，二値化が挙げられる [6]．この研究では，CNN の入出力や重みを，+1 または −1 の二値に近似することで，CNN 内の乗算を単純な XNOR 演算で置き換えている．この考え方を応用し，多数の XNOR 演算ユニットを備えた CNN アクセラレータが提案されており，回路面積や消費電力を小さく抑えることができる [4]．しかし，学習によって獲得した CNN のパラメタを二値に近似して使用するため，CNN の認識精度は有意に低下してしまい，アプリケーションによってはこれが無視できない場合がある．

以上で述べたように，CNN が持つパラメタの近似化によって計算量を削減することができ，アクセラレータを単

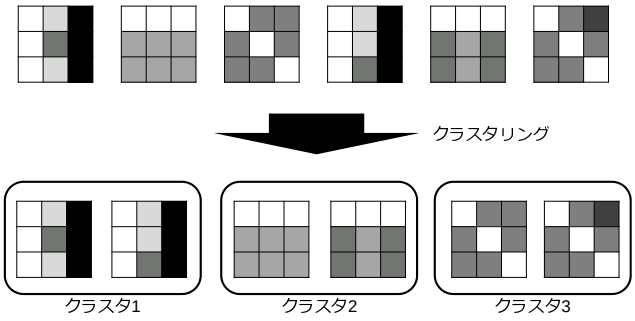


図 3 カーネルクラスタリング

純な回路で構成することが可能となるが，近似方法によっては，CNN の認識精度が大きく低下してしまう．このように，計算の削減量と CNN の認識精度はトレードオフの関係にあり，これら 2 つをできるだけ両立できる近似方法が望ましい．本稿では，CNN の認識精度を保ちつつ，高い電力効率を達成するアクセラレータの実現を目指し，CNN のパラメタを適切に近似する方法を探る．

2.3 カーネルクラスタリング

2.2 節で述べたように，近似方法によっては，元の CNN のパラメタが大きく変更され，認識精度が低下してしまう．そこで，認識精度を保ちつつ，計算量やパラメタのデータ量を削減可能な近似方法を検討する．CNN の畳み込み層に着目すると，一般に，異なるネットワークにおける異なるデータセットに対する学習で獲得されたカーネル同士でも，類似の特徴を示すことがある．そのため，この性質を応用してカーネルをいくつかのクラスタに分類することが可能である．このカーネルクラスタリングを CNN の計算量削減に応用した研究が行われている．例えば，Emily ら [5] は，類似したカーネルを 1 つのカーネルと見なし，カーネル数を減らすことで，畳み込み層に必要な計算量の削減を図っている．そのためにまず，全てのカーネルをクラスタリングし，カーネルをいくつかのクラスタに分類する．このカーネルクラスタリングの例を**図 3** に示す．図 3 では，6 個のカーネルをクラスタ 1〜3 に分類している．その後，クラスタ 1〜3 のそれぞれに含まれる 2 個のカーネルを 1 個のカーネルとして扱い，1 度の畳み込み演算の結果を 2 度利用することで計算量を削減する．

以上で述べた，カーネルクラスタリングに基づく CNN の近似計算の考え方を，ハードウェアアクセラレータに応用する方法を本稿では検討する．具体的には，同クラスタに分類されたカーネルの計算を共通の近似計算に置き換えることで，類似した複数カーネルの計算を 1 種類の計算にまとめ，さらにその計算を専用回路に担当させる．カーネルの畳込み演算を専用回路に置き換える例を**図 4** に示す．この例では，図 3 のクラスタリング結果を基に，カーネルの畳み込み演算を専用回路に担当させている．クラスタリングによって 6 個のカーネルが 3 クラスタに分類されているため，クラスタ 1〜3 に対応する専用回路 1〜3 を用意する．それぞれの専用回路は，対応したクラスタに含まれるカーネルの畳み込み演算の結果を再現する働きを持つ．つまり，入力マップを専用回路 1 に入力することで，クラスタ 1 に含まれるカーネルを入力マップに畳み込んだ場合に得られる出力マップに類似した，出力マップを得ることができる．このようにカーネルの畳み込み演算を専用回路に割り当てることで，積和演算やカーネルデータの移動も削減でき，アクセラレータの電力効率を向上させることが可能であると考えられる．

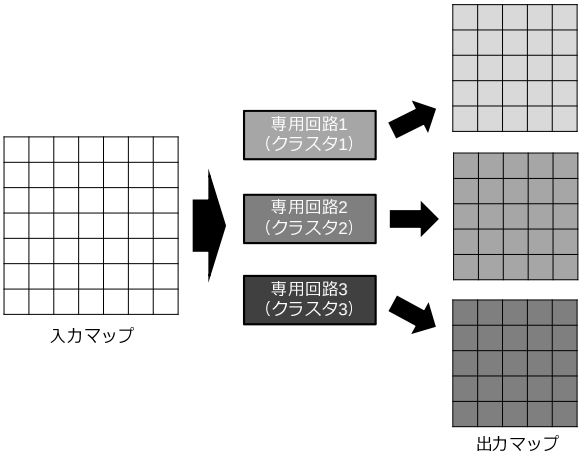


図 4 専用ハードウェアによる畳み込み演算

3. CNN カーネルのクラスタリング評価

2.3 節で述べたようなカーネルの近似化をアクセラレータに応用するために，類似した特徴を示すカーネルが様々な種類の CNN の，複数の畳み込み層中に存在することを確認する．さらに，各クラスタに含まれるカーネルの計算を 1 種類の近似計算に代替する方法，および近似計算による認識精度の変化について考察する．

3.1 評価環境

評価では，ディープラーニングフレームワークである Chainer[7] 上で，CNN における推論処理を実行する．異なる種類の CNN に共通して，類似したカーネルが存在することを確認するために，現在の主要な CNN から VGG16[8] と GoogLeNet[9] の 2 種類を使用した．これらの CNN 内で使用した畳み込み層のカーネルサイズ，枚数を**表 1** に示す．なお，それぞれの層には，1 チャネルごとに表に示した枚数のカーネルが存在する．つまり，VGG16 の Conv1 には，サイズ 3×3 のカーネルが，$64 \times 3 = 192$ 個存在することになる．また，CNN の認識精度への影響を調査する際に，データセットとして ImageNet[10] を用いた．このデータセットを使用した学習によって獲得されたカーネルを，k-means アルゴリズム [11] を用いてクラスタリングした．その際には，全カーネルの値を正規化し，k-means ア

表 1　カーネルの枚数およびサイズ

	VGG16			GoogLeNet		
	枚数	チャネル	サイズ	枚数	チャネル	サイズ
Conv1	64	3	3 × 3	64	3	7 × 7
Conv2	64	64	3 × 3	192	64	3 × 3

ルゴリズムの入力データとして使用する．なお，k-means アルゴリズムは，事前にクラスタ数を指定する必要があるが，今回は各クラスタに属するカーネル数のばらつきができるだけ均等になるような値を経験的に決定した．

3.2　評価結果

カーネルクラスタリングの結果を**図 5** に示す．クラスタ数は 30 に設定しており，1 つのクラスタに含まれるカーネルを横方向に並べ，縦方向にクラスタ番号と共に並べて示している．なお，このカーネル可視化にあたって，各要素値を，クラスタリング対象の全カーネル内に含まれる要素の最大値と最小値によって正規化しており，画素の色が白いほど大きな値，黒いほど小さな値を表している．

まず，1 つの CNN 中で複数の畳み込み層に共通して，類似した特徴を持つカーネルが存在することを確認する．図 5(i) は，VGG16 の Conv1 と Conv2 の 1 つのチャネルに含まれるカーネルをまとめてクラスタリングした結果を表している．これを見ると，Conv1 のカーネルと Conv2 のカーネルが両方とも属するクラスタが存在することが分かる．例えば，クラスタ 6 に着目すると，Conv1 のカーネルが 2 枚，Conv2 のカーネルが 1 枚分類されている．これらのカーネルは，左上に白色の画素を含んでいる点で類似していることが分かる．このことから，複数の畳み込み層に共通して，類似したカーネルが存在していることが確認できる．

次に，異なる種類の CNN にも共通して，類似したカーネルが存在することを確認する．図 5(ii) は，VGG16-Conv1 と GoogLeNet-Conv2 の 1 つのチャネルに含まれるカーネルをまとめてクラスタリングした結果を表している．これを見ると，VGG16-Conv1 のカーネルと GoogLeNet-Conv2 のカーネルが両方とも属するクラスタが存在することが分かる．例えば，クラスタ 18 に着目すると，VGG16-Conv1 のカーネルが 2 枚，GoogLeNet-Conv2 のカーネルが 1 枚分類されており，これらのカーネルは中央縦方向に白色の画素が並んでいる点で，互いに類似していることが分かる．このことから，異なる種類の CNN に共通して，類似したカーネルが存在していることが確認できる．

3.3　考察

3.3.1　カーネルの近似方法

3.2 節で述べたように，様々な CNN が持つ畳み込み層に共通して，類似したカーネルが存在する．そのような類似したカーネルの畳み込み計算を近似計算に置き換える方法について考察する．

カーネルの畳み込み計算を近似計算に置き換える際には，近似による計算結果への影響を最小限に抑えるのが望ましい．そこで，各クラスタ内のカーネルが持つ特徴を適切に表す近似カーネルをそれぞれ生成し，この近似カーネルによって各クラスタ内のカーネルを置き換える．本稿では，各クラスタの近似カーネルを，同一クラスタに属するカーネルを平均化することで求める．この平均化によって，同一クラスタに属するカーネル全ての特徴を考慮した近似カーネルの生成を図る．

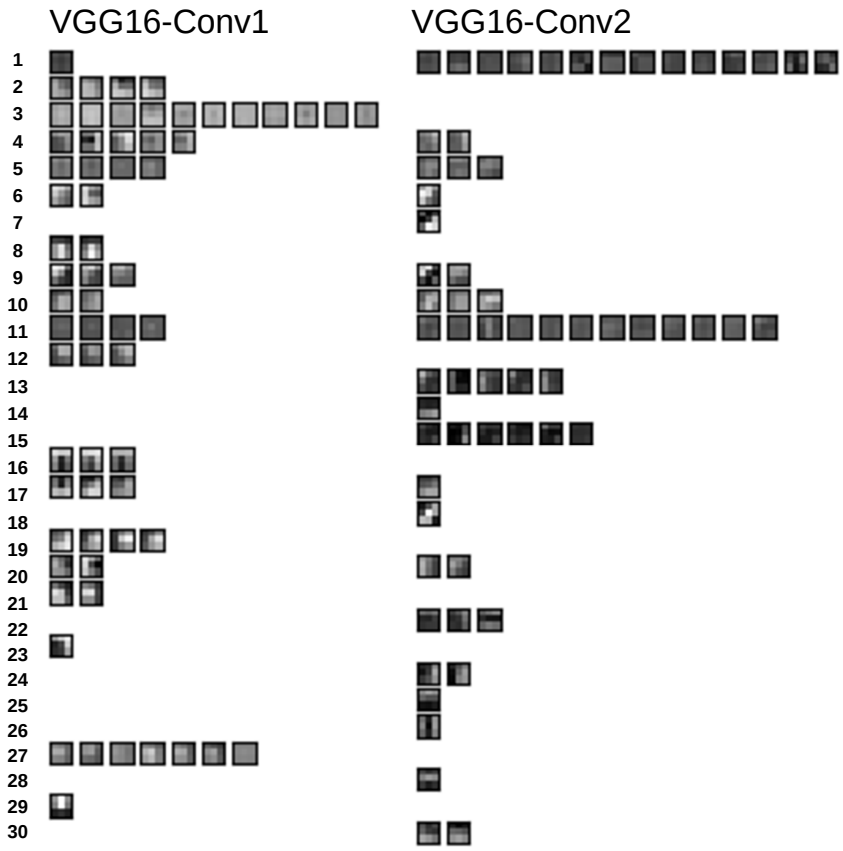
3.3.2　カーネルの近似化による影響

本稿では前項で述べたような方法を用いて，各クラスタのカーネルの畳み込み演算を近似計算に置き換えた場合の，CNN の認識精度に与える影響を確認する．認識精度の算出には，ImageNet の画像と，画像のラベルのセットを 1000 枚使用した．この画像を CNN に入力し，1000 個のカテゴリに識別させ，正しく識別できた割合を求めた．なお，CNN が出力した結果の第 1 候補が正解となる割合である Top-1 Accuracy と，第 5 候補までに正解が含まれる割合である Top-5 Accuracy をそれぞれ求めた．元の CNN と，カーネルを近似した CNN の両方で認識精度を求め，その結果を比較することで，カーネルの近似化が CNN の認識精度に与える影響を確認した．

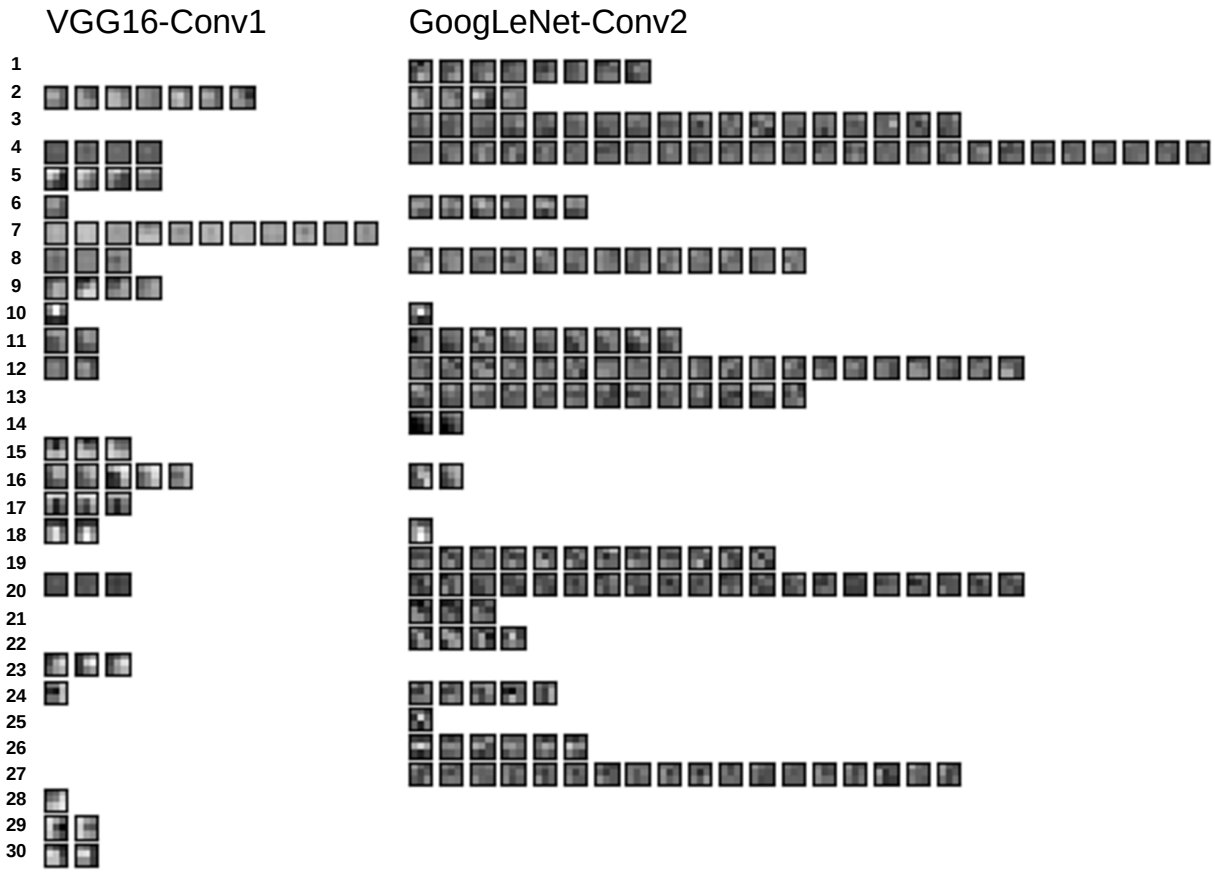
今回は，VGG16 の Conv1 に含まれる 192 個のカーネルを近似化の対象とする．これらのカーネルを 30 個のクラスタに分類し，各クラスタに属するカーネルを近似カーネルで置き換える．また，近似の対象とするクラスタの数と，認識精度の関係を調べるために，近似の対象とするクラスタを 1 個〜30 個の範囲で変化させて，Top-1 Accuracy と Top-5 Accuracy の変化を調査した．なお，近似の対象とするクラスタの選択方法によって，認識精度の変化の仕方が異なると考えられるため，次の 3 つの方法で近似対象のクラスタを選択した．

(**A**) クラスタに属するカーネル数が少ない順に選択
(**B**) クラスタに属するカーネルの各要素の絶対値の平均値が小さい順に選択
(**C**) クラスタに属するカーネルの各要素の分散が小さい順に選択

1 つ目の選択方法は，近似するカーネルの数が少ない程，認識精度の低下量も小さいという考えに基づいている．2 つ目の選択方法は，各要素が 0 に近いカーネル程，認識精度への寄与度が小さいという考えに基づいている．カーネルは畳み込み演算の際に，入力マップとの積を計算するため，カーネルの各要素の絶対値が小さいほど，出力マップの絶対値も小さく，ネットワーク全体に対して影響力が小さいと考えられる．そのため，各要素が 0 に近いカーネルを近似したとしても，認識精度の低下量は小さいと考えら



(i) VGG16 の Conv1 と Conv2



(ii) VGG16 の Conv1 と GoogleNet の Conv2

図 5 カーネルクラスタリング結果

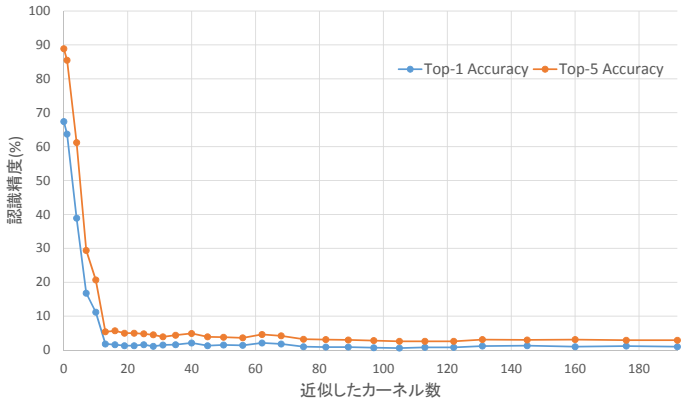


図 6 選択方法 (A) によって近似するクラスタを選択した場合の認識精度の変化

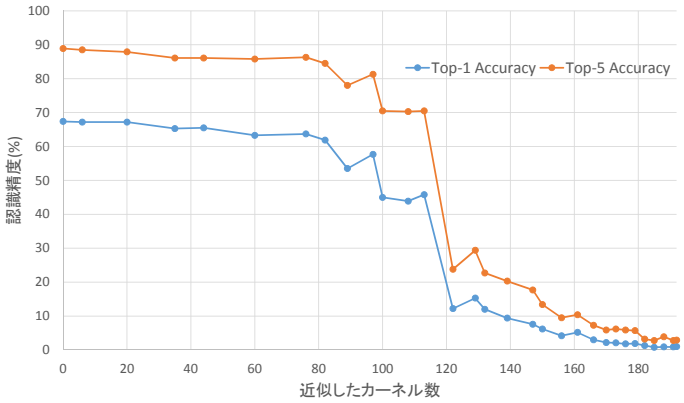


図 7 選択方法 (B) によって近似するクラスタを選択した場合の認識精度の変化

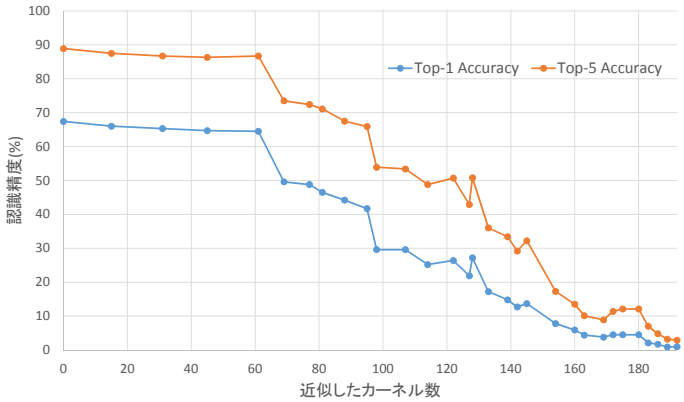


図 8 選択方法 (C) によって近似するクラスタを選択した場合の認識精度の変化

れる．3 つ目の選択方法は，各カーネルのばらつきが小さいクラスタ程，近似して 1 つのカーネルにまとめても誤差が小さいという考えに基づいている．この方法では，クラスタに属するカーネルの要素ごとの分散を求め，その分散の平均を求める．この分散の平均が小さいほど，クラスタに属するカーネルのばらつきが小さく，近似したとしても認識精度の低下量は小さいと考えられる．以上で述べた 3 つの方法で近似の対象とするクラスタを選択し，認識精度の変化を調査した．

まず，選択方法 (A) によって近似するクラスタを選択した場合の認識精度の変化を**図 6** に示す．図 6 では，認識精度の変化を 2 つの折れ線グラフで示しており，青色が Top-1 Accuracy，オレンジ色が Top-5 Accuracy をそれぞれ表している．グラフの縦軸が認識精度，横軸が近似したカーネルの数をそれぞれ表している．また，グラフ上の 31 個の点が，近似したクラスタの数ごとの認識精度を表している．つまり，左の点から順にそれぞれ 0〜30 個のクラスタに属するカーネルを近似した場合に対応している．結果を見ると，カーネルの近似により，認識精度が大きく低下してしまっていることが分かる．特に，左から 6 個目の点，つまり 5 個のクラスタに属するカーネルを近似した時点で，Top-5 Accuracy が約 5%まで低下している．元の CNN の Top-5 Accuracy が 90%近くあることを考えると，実用的ではない認識精度であることが分かる．

次に，選択方法 (B) によって近似するクラスタを選択した場合の認識精度の変化を**図 7** に示す．結果を見ると，選択方法 (A) の結果とは大きく異なり，7 個のクラスタに属する約 80 個のカーネルを近似しても，Top-5 Accuracy を 80%以上に保つことができていることが分かる．このことから，クラスタによって，近似した際の認識精度への影響が大きく異なること，また選択方法 (B) は選択方法 (A) と比較して，認識精度への寄与度が小さいクラスタから順に近似できていることがうかがえる．

最後に，選択方法 (C) によって近似するクラスタを選択した場合の認識精度の変化を**図 8** に示す．認識精度の変化に着目すると，約 60 個のカーネルを近似したあたりで，認識精度が大きく低下しており，今回の環境では選択方法 (C) よりも選択方法 (B) のほうが，認識精度への影響が小さいクラスタから順に選択できたと考えられる．一方で，クラスタ数とカーネル数の関係に着目すると，認識精度が大きく低下する直前では，4 個のクラスタに属する約 60 個のカーネルを近似した上で，80%以上の Top-5 Accuracy を保てている．選択方法 (B) では，60 個のカーネルを近似するために，5 個のクラスタの近似を必要としているため，選択方法 (C) は選択方法 (B) よりも少ないクラスタで，多くのカーネルを近似可能であることが分かる．

以上のことから，認識精度に大きく寄与しているクラスタと，そうでないクラスタが存在することが確認できた．また，認識精度への寄与度が小さいクラスタから順に近似対象として選択していくことで，認識精度の低下が抑えられることを確認した．

4. クラスタリングを応用するアクセラレータの構想

以上で述べたように，様々な CNN が持つ層には，類似した特徴を示すカーネルが多数存在し，それらカーネルの畳み込み演算を 1 つの近似計算に置き換えることが可能で

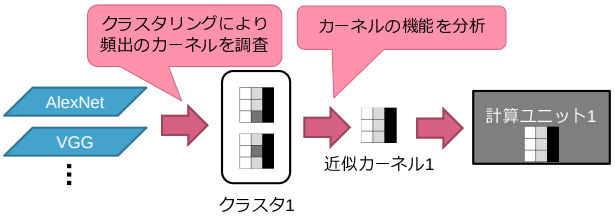
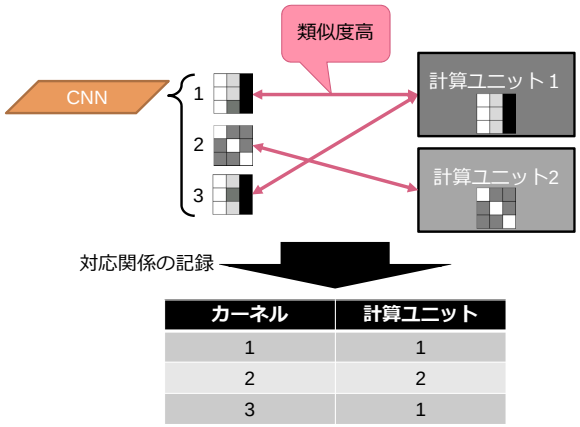


図 9　近似するクラスタの決定と計算ユニットの設計



カーネル	計算ユニット
1	1
2	2
3	1

図 10　計算ユニットで代替するカーネルの決定

あると考えられる．本稿ではこの性質を利用する CNN アクセラレータについて検討する．

4.1　設計方針

本稿で検討するアクセラレータは，様々な CNN に共通して含まれるカーネルの畳み込み演算を近似的に行う計算ユニットを複数持ち，学習済み CNN における推論処理を近似的に実行する．アクセラレータに搭載する計算ユニットは，様々な CNN のカーネルクラスタリングの結果に基づいて設計する．計算ユニットを設計する流れを，**図 9** を用いて説明する．例えば，様々な CNN をクラスタリングした結果，多くのカーネルが属するクラスタ 1 の存在が確認できたとする．このクラスタに含まれるカーネルを近似計算の対象とし，クラスタ 1 に属するカーネルが持つ機能を分析する．分析の結果，クラスタ 1 には「縦方向のエッジを検出する働き」を持つカーネルが多く含まれていると判明したとする．この場合，「縦方向のエッジを検出する働き」を持つ近似カーネル 1 を作成する．その後，この近似カーネル 1 の畳み込み演算を高効率に行う計算ユニット 1 を設計し，アクセラレータに搭載する．これを近似計算の対象となったクラスタに対して繰り返していき，アクセラレータに搭載する全ての計算ユニットを設計する．

そして，このアクセラレータを用いて学習済みの CNN における推論処理を実行する際には，搭載している計算ユニットの近似カーネルと，各カーネルとの類似度を調査し，各計算ユニットで計算を代替するカーネルを事前に決定しておく．計算ユニットに担当させるカーネルを決定する様子を**図 10** に示す．推論対象の学習済み CNN が与えられると，CNN 内のカーネルを取り出し，全ての計算ユニットが持つ近似カーネルとの類似度を調査する．そして，類似度が閾値を上回った場合，その計算ユニットにカーネルの計算を担当させることを決定し，カーネルと計算ユニットの対応関係を記録していく．図 10 の例では，カーネル 1 とカーネル 3 は，計算ユニット 1 が持つ近似カーネルとの類似度が高く，「縦方向のエッジを検出する働き」を持っていると判断される．そのため，カーネル 1 とカーネル 3 の計算は，計算ユニット 1 で代替可能であると考えられるため，その対応関係を記録する．このようにして得られた，カーネルと計算ユニットの対応関係を事前にアクセラレータに転送しておくことで，アクセラレータでの推論処理が実行可能となる．

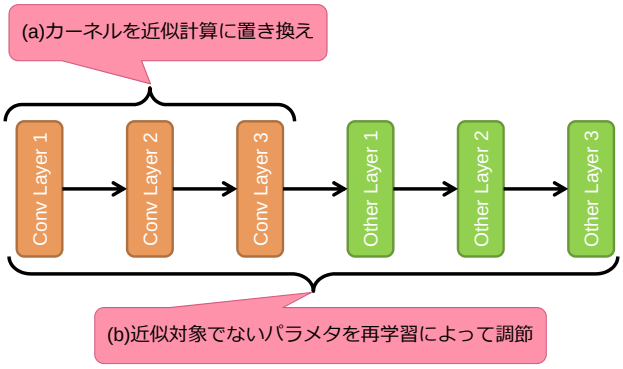


図 11　ネットワークの再学習

しかし，カーネルの畳み込み演算を近似計算に置き換えただけでは，推論時の認識精度が低下する可能性がある．そこで，カーネルの畳み込み演算を近似計算に置き換えた後，ネットワークを再学習することで，認識精度の低下を抑制する．ここで，**図 11** に示す CNN を用いて再学習の方法を説明する．この CNN は 6 層で構成され，そのうちの上位 3 層に畳み込み層が位置している．この 3 層の畳込み層を近似計算の対象として，CNN における推論処理をアクセラレータ上で実行する場合を考える．推論対象として CNN が与えられると，3 層の畳込み層内のカーネル計算が近似計算に置き換えられる (a)．この状態では，値が変更されたパラメタと，そうでないパラメタが混在することとなり，CNN の認識精度が低下してしまうと考えられる．そのため，カーネルの畳み込み演算を近似計算に置き換えた状態で，近似の対象とならなかったカーネルと畳み込み層以降の層を対象に再学習を行う (b)．この再学習によって，元の CNN の認識精度に近づくようにパラメタを調整する．以上で述べた方法によって，CNN の認識精度の低下を抑制しつつ，カーネル計算を近似計算に置換する．

4.2　構成概要

以上で述べた方針に従い，カーネルクラスタリングを応

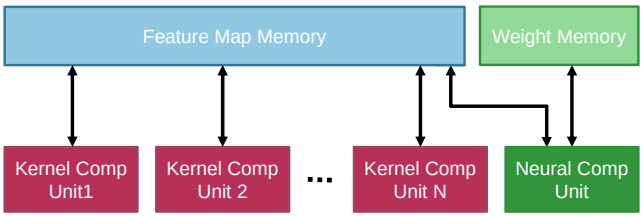


図 12　アーキテクチャの構成

用するアクセラレータを設計した．図 12 にアクセラレータの構成を示す．アクセラレータは，複数の近似計算ユニット (Kernel Comp Unit) と，積和演算ユニット (Neural Comp Unit) から構成される．全てのユニットは Feature Map Memory と接続しており，これに対して入力マップや出力マップを読み書きする．また，積和演算ユニットのみ Weight Memory と接続しており，ここからカーネルを読み出すことができる．このアクセラレータでは，事前に転送された，カーネルと計算ユニットとの対応関係に基づき，カーネルの畳込み演算を近似計算ユニットに割り当てる．なお，対応する近似計算ユニットが存在しないカーネルおよび畳込み層以外の層の処理は，積和演算ユニットが担当する．推論時には，Feature Map Memory を介して，ニューロンの入出力データが全てのユニットで共有され，ニューロンの出力を各ユニットへ入力することで計算を進めていく．

4.3　考察

このアクセラレータでは，搭載する近似計算ユニットの数が消費電力および計算時間に影響を与える．近似計算ユニットの数が増加すると，多くのカーネルの種類に対応することができるため，消費電力の大きい積和演算ユニットを使用する回数を減らすことができ，消費電力も削減されると考えられる．さらに，同時に計算可能なカーネルの数が増えるため，計算時間が短縮されると考えられる．しかし，近似計算ユニットの増加に伴い，リーク電流および回路面積が増大してしまう．

したがって，近似計算ユニットの数は，回路面積や計算時間を考慮して決定する必要がある．また，多くのカーネルを置き換え可能な近似計算ユニットを優先的に搭載することで，近似計算ユニットの数を少なく抑えることができると考えられる．

5.　おわりに

本稿では，高電力効率な CNN アクセラレータの実現を目指し，CNN に含まれる計算を適切に近似する方法を検討した．特に，CNN の特徴であるカーネルの畳み込み演算に着目し，カーネルをクラスタリングすることで，クラスタ単位でカーネルの畳込み演算を近似する方法を検討した．VGG や GoogleNet など，現在広く使用されている CNN のカーネルをクラスタリングした結果，様々な CNN が持つ畳み込み層には，類似したカーネルが多数存在することを確認した．また，クラスタごとに CNN の認識精度への寄与度が異なり，近似対象とするクラスタの選択方法によって，認識精度の低下量が大きく異なることを確認した．

この結果に基づき，本稿では近似計算ユニットを複数備えたアクセラレータの設計を検討した．また，CNN のカーネルの畳み込み演算を近似して計算する際に，再学習を行うことで，認識精度の低下を抑制する方法を検討した．このアクセラレータによって，積和演算ユニットを使用する回数を削減でき，CNN における推論処理を実行する際の電力を削減できると考えられる．

今後の課題として，カーネルの畳込み演算を近似する方法および，認識精度の低下を抑えることができるクラスタ選択方法の検討が挙げられる．また，電力効率や計算時間を考慮した場合の，アクセラレータに搭載する近似計算ユニットの適切な数や種類に関する調査が必要である．

参考文献

[1] Chen, T. et al.: DianNao: a small-footprint high-throughput accelerator for ubiquitous machine-learning, *Proc. 19th Int'l Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS'14)*, pp. 269–284 (2014).
[2] Chen, Y. et al.: DaDianNao: A Machine-Learning Supercomputer, *Proc. 47th Annual IEEE/ACM Int'l Symp. on Microarchitecture (MICRO-47)*, pp. 609–622 (2014).
[3] Chen, Y.-H., Emer, J. and Sze, V.: Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks, *Proc. 43rd Annual Int'l Symp. on Computer Architecture (ISCA'16)*, pp. 367–379 (2016).
[4] Nakahara, H., Yonekawa, H., Sasao, T., Iwamoto, H. and Motomura, M.: A memory-based realization of a binarized deep convolutional neural network, *Field-Programmable Technology (FPT), 2016 International Conference on*, IEEE, pp. 277–280 (2016).
[5] Denton, E. L., Zaremba, W., Bruna, J., LeCun, Y. and Fergus, R.: Exploiting linear structure within convolutional networks for efficient evaluation, *Advances in Neural Information Processing Systems*, pp. 1269–1277 (2014).
[6] Rastegari, M., Ordonez, V., Redmon, J. and Farhadi, A.: Xnor-net: Imagenet classification using binary convolutional neural networks, *European Conference on Computer Vision*, Springer, pp. 525–542 (2016).
[7] Tokui, S., Oono, K., Hido, S. and Clayton, J.: Chainer: a next-generation open source framework for deep learning, *Proceedings of workshop on machine learning systems (LearningSys) in the twenty-ninth annual conference on neural information processing systems (NIPS)* (2015).
[8] Simonyan, K. and Zisserman, A.: Very deep convolutional networks for large-scale image recognition, *arXiv preprint arXiv:1409.1556* (2014).
[9] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S.,

Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A.: Going Deeper with Convolutions, *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9 (2015).

[10] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C. and Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge, *International Journal of Computer Vision (IJCV)*, Vol. 115, No. 3, pp. 211–252 (online), DOI: 10.1007/s11263-015-0816-y (2015).

[11] MacQueen, J. et al.: Some methods for classification and analysis of multivariate observations, *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1, No. 14, Oakland, CA, USA., pp. 281–297 (1967).