輪講資料
＜論文名＞
「SnaPEA: Predictive Early Activation for Reducing Computation in Deep Convolutional Neural Networks」

＜研究背景＞
CNNにおいて、畳み込み演算を行う際、膨大な計算が必要となる。そして、この畳み込み演算の計算結果に活性化関数を適用することで、CNNにおける各層の出力を得ている。
昨今、多く知れ渡っているCNNにはランプ関数と呼ばれる活性化関数が**用いられている**。ランプ関数の概要は、入力値が正の領域において、恒等関数となり、負の領域においては、零の値を返す。
このランプ関数の定義を踏まえると、CNNにおいて、膨大な計算後の負の出力に対して、等しく0を、活性化関数を適用した後の値として出力することとなり、膨大な計算の意味が損なわれる。
本論文では、SnaPEAと呼ばれるデバイスを開発し、負の出力を早期に検知することで、計算量の削減を達成した。

＜SnaPEAについて＞
SnaPEAの概要を以下に示す。
- SnaPEAはCNNのアルゴリズム構造を利用して計算量を削減する。
- SnaPEAにより、CNNの計算のランタイムが短くなる。
- SnaPEAは、精度のトレードオフを制御するためのハードウェアとソフトウェアの側面から問題を解決する。

そして、SnaPEAには正確モードと予知モードの二つが存在する
正確モードの場合、分類精度を損なうことなく、負の部分和の計算の早期終了を引き起こす。これに計算量を削減する。
予知モードの場合、畳み込み演算結果と適切なしきい値を比較することにより、正確モードより早期に負を検知し、大きな計算量を削減できる。しかし、計算量の削減と引き換えに分類の精度が小さくなる。

＜正確モードでのソフトウェアの流れ＞
ReLu関数により、畳み込み演算において、正のカーネルを用いた計算後の出力は正に保たれる。つまり、負のカーネルを用いた畳み込み演算においてのみ、負の出力を持つ。またこのような演算をMAC演算とよぶ。
この考えに基づいて、正確モードでは、符号ベース重み再順序化パスにおいて、畳み込みカーネルの重みを符号に基づいて再順序化し、正のサブセットが負のサブセットの後に続くよう並び替える。この並べ替えにより、SnaPEA は最初に正のサブセットで MAC を実行するその後。負の重みを持つ計算中に負の部分出力が観測された場合に、計算を早期終了することで活性化関数をより早く適用する。

＜予知モードでのソフトウェアの流れ＞

予知モードにおいて、負の重みでの演算を始める前に、しきい値により演算を行うかを制御する。直感的には、一定数のMAC演算後、畳み込み演算の部分出力が閾値未満であれば、最終的な畳み込み出力は負の値になる可能性が高いと予想し計算を早期終了する。

この流を実現するためには、しきい値の決定、そしてそれに関係するMAC演算数を決定する必要がある。

この二つのパラメータの決定は3つの方法から決定することができる。

始めに、各カーネルのパラメータ、その次に、各層のパラメータ、最後に、CNNにおけるパラメータ、つまり、目的とするしきい値、MAC演算数が設定される。

この二つのパラメータを決定するにあたり、論文５P、Algorithm1が用いられる。詳しい決定方法については後の章で説明を記述する。

得られた全体の推測パラメータに基づいて、各カーネルの重みを重み再順序付けパスによって再順序付けを行う。このパスでは、推測パラメータで決定されたものを他のものよりも優先してカーネルの重みを並べ替える。次に、残りの重みは、符号ベースの重みの並べ替で使用されたのと同じ手順に基づいて並べ替えられ、正の重みの後に負の重みが並べられる。最後に，これらの並べ替えられた重みがSnaPEAハードウェア上でのCNNの実行を決定する.

＜ハードウェア側面からのSnaPEA＞

SnaPEAのProcessing Engine(以下PE)にはカーネル重みを保存する数列保存領域が備え付けられている。この仕組みは、カーネルの重みを並び替える際に、必要となる機能である。

＜正確モードにおけるハードウェア仕様＞

PEはまず正の重みの演算から行う。負の重みにおいて、部分和が負になると、畳み込み演算を終了する。早期活性化が開始されると、PEは別の畳み込み演算を実行することができる。

＜予知モードにおけるハードウェア仕様＞

各 PE は、あらかじめ定められた数の MAC 演算を行った後、部分和を閾値と比較することで、畳み込み出力の符号を推定する。

＜しきい値とMAC演算数の決定＞

＜重みの決定法＞

予知モードで用いられるしきい値の選択には、正しい重みの選択が不可欠である。

重みを選択する方法の一つとして、重みの絶対値の降順に並べ替え、（推測を行うための一連の操作として）マグニチュードの大きいものを選択することが考えられる。しかし、この方法では正と負の重みを考慮すると、特徴の抽出の精度は劇的に小さくなる。

この影響を軽減するために、SnaPEAでは、重みを昇順に並び替え、いくつかの小さなグループに分割し、各グループの中から最大のマグニチュードを持つ重みを選択する。これにより、比較的小さな重みでも推測のための演算セットに選ばれる。その結果、大きな入力値と結合する可能性のある小さな重みを選ぶことができる。また、この重みを選択する方法を採用することで、ソフトウェアが行うことは重みを分類するグループ数の決定だけすむ。

＜MAC演算数の決定＞

しきい値と重みを分類するグループ数の決定において求められることは、計算量を最大限に削減し、その上で、受け入れられる損失を返すことである。つまり、この問題は、最適化問題に落としこむことができる、

MAC演算数の問題を定式化するために、SnaPEAによって実行されるMAC演算の数を、変更されていないCNNによって実行されるMAC演算の数から差し引くことによって、計算量の削減を測定する。しかし、変更されていないCNNのMAC演算の数は様々な入力に対して一定であるため、計算量削減を最大化することは、SnaPEAが実行するMAC演算の数を最小化することと等価である。

ここで、ある層ｌで、カーネルｋ、入力イメージｄの際のMAC演算数を決定する。

MAC演算数の決定は以下で考えを用いている。

1. 部分和がしきい値以下であれば、重みを選んだ際のグループ数がMAC演算数
2. 部分和をがしきい値以上、重みが負であれば、負の重みのindexがMAC演算数
3. それ以外であれば、合計のカーネル数がMAC演算数数。合計のカーネル数は入力チャネル数＊カーネルサイズ＊カーネルサイズで求められる。

すべての畳み込みを生成するための計算量の出力は、個々の出力を生成するのに必要なMAC操作の数の合計である。この定義に基づいて、問題は、MAC操作の総数を最小化し、精度損失に関する制約を満たす推測パラメータを見つけることに変換される。式は論文P5、（2）を参照

＜アルゴリズムを用いたパラメータの決定＞

このアルゴリズムでは，まず，各カーネルで実行される推測に対するCNNの感度を特徴づける．そして，損失を受け入れる値であるε以下に抑えながら計算量を最小化するように，すべてのカーネルの推測パラメータを探索して調整する．したがって，本アルゴリズムは以下のように大きく2つの段階（プロファイリング段階と最適化段階）に分けられる．

＜プロファイリング段階＞

アルゴリズム 1 の関数 KernelProfilingPass は、レイヤ l のカーネル k について、パラメータ の様々な値に対応する演算数（op）と精度損失（err）をプロファイリングする。しきい値th、重みを分類するグループ数ｎとし、（th, ｎ）の値の一つとして(0,1)を設定することで，各カーネルの正確モードもプロファイリング結果に含まれる．この処理はCNN内のすべてのカーネルに対して繰り返される．精度損失の観点から許容できるプロファイリング結果は，ParamKと呼ばれるリストに蓄積される．リストParamKの各サブリストParamK[l][k]はopの値に基づいて昇順に並び替えられる。

＜最適化段階＞

この段階では、最適なパラメータが設定される。そしてこの段階はLocalOptimizationPassとGLobalOptimizationPassから構成される。詳しいアルゴリズムについては論文P5、Algorithm1 を参照。

プロファイリング段階で並び替えられたParamkをもちいて、LocalOptimizationPassではCNN内の各層にたいしてある一定数のパラメータの候補のリストを作成する。パラメータは最適な命令数と、許容できる損失の条件を満たした場合にリストに追加される。ある一定数のパラメータの候補が集まった時、そしてこのリストが、GlobalOptimizationPassに応用させる。

GlobalOptimizationPassでは、パラメータの候補をもとに、分類精度と計算量削減に対する層間効果を考慮して推測パラメータを調整し、最終的なパラメータを決定する。また、このパラメータの選定の際、損失あたりの命令数で除算することで、命令の有効度をはかる。この考えを用いることにより、より少ない命令数、つまり計算量の削減とより小さな損失を達成し適切なパラメータをLocalOptimizationPassで選ばれたリストから選定する。

＜SnaPEAの構成について＞

SnaPEAアーキテクチャは、ReLU活性化層とプーリング層の計算をサポートする専用ユニットで構成されている。

＜Proccesing Engineの構成について＞

PEの特徴を以下に示す。

- PEそれぞれは、複数の計算の領域を持ち、さらに重み、インデックス、そして入力、出力保存領域を持つ。
- それぞれの計算領域はある一つのMAC演算が占領することができる。
- それぞれのPEにおいて、重み、インデックス、そして入力、出力保存領域は複数の計算領域をこえて共有されている。
- 各ＭＡＣユニットは、畳み込み演算の際は、１つの入力と重みの乗算を行い、その結果をアキュムレーションレジスタに送る。
- 予測活性化ユニット（PAU）は、部分和の値をチェックして、各畳み込み計算についてさらなる計算が必要かどうかを判断する。PAU が畳み込みウィンドウの追加計算が必要ないと判断した場合、エネルギーを節約するために、対応する乗算器とアキュムレータをデータが送られる。プロセスは、現在の畳み込み計算のすべての計算が実行されるか、またはPAU が早期に活性化関数を適用することを決定するまで継続される。

＜Prediction Activation Unitの構成について＞

畳み込み演算を補助するため、PEにおいて、それぞれの計算領域にPAUが備え付けられている。また、PEの構造は論文P8,図７を参照。

正確モードで畳み込み演算を実行することは、負の重みを持つＭＡＣ演算の間に部分和値の符号をチェックすることのみ必要とされる。従って、正確モードでは、信号Ｐｒｅｄｉｃｔはゼロに設定され、これにより、レジスタＡｃｃ　Ｒｅｇに格納された部分和値の符号ビットが畳み込み演算の終了を決定する。符号ビットが1 になると、信号 terminate がアサートされ、基礎となる畳み込み計算の残りの計算を終了するようにコントローラに通知される。

予測モードでは、畳み込み出力の符号は、ソフトウェア部分を通して静的に決定される閾値(th)とそれに関連する演算数(n)を通して推測される(アルゴリズム1参照)。推測を実行するために、PAUはまず、アキュムレータレジスタからの部分和値を、予め定められた数のMAC演算の後にしきい値で比較を行う。このとき、コントローラは、信号Ｐｒｅｄｉｃｋｔを１に設定する。部分和値が予め定められた閾値よりも小さい場合、ＰＡＵは、この畳み込み窓の最終的な値が最終的に負になる結果を返す。この場合、PAU は以下のタスクを実行する。(1) この畳み込みウィンドウに対してそれ以上の計算が必要ないことをコントローラに通知し、(2) 早期のReLU活性化を実行してゼロを出力バッファに送信する。計算領域において、負の重みに達するまで畳み込みウィンドウの計算を通常通り継続し、部分和の次のチェックは、負の重みでMAC演算を開始した時点で開始される。部分和の値が所定のしきい値よりも大きい場合、信号Ｐｒｅｄｉｃｔはデアサートされ、ＰＡＵは、各ＭＡＣ演算の後、部分和値に対して、正確モードで述べたプロセスと同様に、定期的に単純な１ビットの符号チェックを実行する。符号ビットが１になると、ＰＡＵは現在のウィンドウの畳み込み演算を終了し、ゼロ値を出力に送る

＜論文対する疑問点＞

学習済みのネットワークに対してしか、このシステムはもちいることができないのではないか。CNNの学習において、重みつまり、カーネルの値は常に更新されていく。このことを考慮すると、カーネルについて特徴を分類するKernelProofilingPassにおいて、シミュレーションを行い、推定に必要なパラメータを決定していく。シミュレーションをおこなうことは、結果として演算をする際、計算量を削減することなく行っていることと同様であるため、学習中のネット―ワークに用いることはできないと考える。

Algorithm1 において、op、errを決定する際、Simulateと呼ばれる関数が用いられているが、このSimulateについて詳しい記述がない。