ISSN 2186-5647

2-30

自己増殖型連続畳み込みニューラルネットワーク

日大生産工　○小林 勇太　　日大生産工　山内　ゆかり

1. まえがき

連続畳み込みは不規則にサンプリングされたデータを扱い、長期的な依存性をモデル化できることから注目を浴びている[1] [2]。また、大きな畳み込みカーネルを使用した実験で成功を収めて以来、大きなカーネルを効率的に構築できる連続畳み込みの開発が盛んになった。連続畳み込みを実装するためのアプローチとして多層パーセプトロン(MLP)が活用されているが、計算コストの高さやハイパーパラメータの設定の複雑さ、フィルタの記述力の限界など、いくつかの欠点が存在する。これらの問題を解決するためにSanghyeon KimらはSMPConvと呼ばれる連続畳み込みのための自己移動点表現[3]を提案し、少ない計算コスト、少ないパラメータ数で性能の向上を実現させた。また、大規模な設定(ImageNet)における連続畳み込みの有効性を初めて実証した。しかし、実験データごとに最適な誘導点数を決定することが難しく、また、各フィルタの誘導点座標と数が共通のため、各フィルタを最適化することが困難である。

本研究では、学習精度の向上と計算量の最適化を図るために、学習によって誘導点数を自動的に増減させるアルゴリズムを提案する。誘導点の数は学習精度と計算量に影響を大きく与えるため、誘導点の数を自動で学習させることで、誘導点の数をフィルタごとに最適化させ、精度の向上と計算量の最適化を図る。比較実験は画像分類タスクで行い、提案手法とSMPConvの比較と、提案手法ごとの比較をし、学習精度及び計算速度について報告する。

2. 従来研究

連続関数を表現するために自己移動点表現を用いる。入力座標の次元の大きさをd(時系列データは1、画像データは2)、チャンネル数をN_Cとしたとき、ベクトル値関数は$\mathrm{SMP}: R^d \to R^{N_C}$となる。これは、入力座標から出力カーネルベクトルへマッピングすることを意味する。このとき、クエリ点$x \in R^d$が与えられたときの連続カーネル関数を以下のように定義する。

$$\mathrm{SMP}(x;\varphi) = \frac{1}{|N(x)|}\sum_{i\in N(x)} g(x,p_i,r_i)\,w_i \qquad (1)$$

このとき、$\varphi = \{\{p_i\}_{i=1}^{N_p}, \{w_i\}_{i=1}^{N_p}, \{r_i\}_{i=1}^{N_p}\}$は学習可能なパラメータの集合であり，$N_p$は関数を表現するために用いる点の数である。また、$p_i \in R^d$は自己移動点表現の座標であり、各点表現は学習可能な重み$w_i \in R^{N_C}$と半径$r_i \in R^+$を持つ。このとき、各レイヤーの各フィルタのカーネルは、それぞれ独自の重み$\{w_i\}$と半径$\{r_i\}$を持っているが、座標$\{p_i\}$は1つのレイヤーで共有している。Figure.1にSMPカーネルの図を示す。

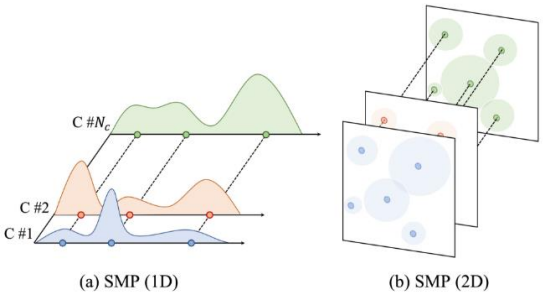

Figure.1　自己移動点表現

$g(x,p_i,r_i)$は距離関数であり、距離関数$g: R^d \times R^d \times R^+ \to R$は次のように定義される。

$$g(x,p_i,r_i) = 1 - \frac{\|x-p_i\|_1}{r_i} \qquad (2)$$

$\|\cdot\|_1$はL1距離である。$N(x)$はクエリ座標xの近傍点のインデックスの集合であり、距離関数gを用いて、$N(x) = \{i | g(x,p_i,r_i) > 0, \forall i\}$として定義される。したがって、ある距離(半径)を超えた点はクエリ点に影響を与えず、SMPは近傍点表現の加重平均によって出力ベクトルを生成する。

式1を利用して連続畳み込み演算を定式化すると下記のようになる($d=1$の場合)。

$$(f * \mathrm{SMP})(x) = \sum_{c=1}^{N_c}\int_R f_c(\tau)\,\mathrm{SMP}_c(x-\tau)\,d\tau \qquad (3)$$

このとき、$f: \mathrm{R} \to R^{N_C}$は入力関数であり、$f_c$と$\mathrm{SMP}_c$は入力の$c$番目の要素を表す。畳み込み演算子は関数を生成し、$N_c$チャンネル全体の合計によってフィルタ応答を計算する。1つのSMP表現は1つの畳み込み演算子に対応し、複数のSMPを使って1つの畳み込み層を実装し、複数の出力チャンネルを生成する。

Self-Organizing Incremental Continuous Convolutional Neural Network
Yuta KOBAYASHI and Yukari YAMAUCHI

3. 提案手法

本研究では、画像分類精度を向上させるために、固定のパラメータとして学習に用いられている誘導点の個数を、学習によって自動で増減させるアルゴリズムを提案する。誘導点の増減は、「誘導点の追加」、「誘導点の削除」、「誘導点の結合」の3つの処理を組み合わせて行う。従来手法では誘導点座標が各フィルタで共通であったが、本研究では各フィルタの誘導点数の最適化を目的とするため、誘導点座標を各フィルタで独立させている。

3.1 誘導点の追加

誘導点の追加は、変化量が最も大きい誘導点の近傍に誘導点を追加する。これは、変化量が大きい誘導点は、特徴量を適切に抽出できるだけの誘導点が近傍に存在していないという仮定のもと、誘導点群が特徴を抽出できるようにすることを目的としている。このとき、過剰に誘導点が追加されることを防ぐために、誘導点を追加するタイミングと変化量の閾値を設定している。また、誘導点が削除された場合、誘導点が過剰に追加されているという仮定のもと、一度でも削除が行われたフィルタでは誘導点の追加ができない仕組みを導入している。追加した誘導点は、既存の誘導点に大きな影響を及ぼさないように、重みと半径は小さな値で初期化し、座標は変化量が最も大きい誘導点の近傍座標で初期化する。また、追加されたノードがすぐに削除されないように、追加してから一定のステップは削除が適用されない仕組みを導入している。誘導点の追加の疑似コードをAlgorithm.1に示し、イメージをFigure.2に示す。

Algorithm.1 Add a Self-moving Point

Constant: AT : 誘導点追加周期, R : 半径の初期値, MAX : 誘導点の最大数

Require: $dmax,new$: 誘導点番号, φ_{new} : パラメータ集合, p_{new} : 追加した誘導点の位置, w_{new} : 追加した各誘導点の重み, r_{new} : 追加した誘導点の半径, $epoc$: 学習回数, np : チャンネルの誘導点数, p_{dmax} : 変化量が最大の誘導点座標, $addW$: 追加した誘導点の重みの初期値

1: if $(epoc == AT)$then
2: if (never deleted && $np < MAX$)then
3: add(φ_{new}) // パラメータ集合を追加
4: $p_{new} \leftarrow$ neighborhood (p_{dmax})
// 変化量が最大の点の近傍座標
5: $w_{new} \leftarrow addW$
6: $r_{new} \leftarrow R$
7: end if
8: end if

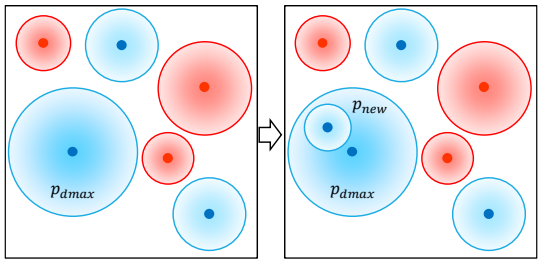

Figure.2 誘導点の追加

3.2 誘導点の削除

誘導点の削除は、影響度が極度に小さい誘導点を削除する。これは、不要な誘導点を削除することで、計算効率を上げることを目的としている。影響度の強弱の判定基準として重みと半径に対して閾値を設定し、閾値を下回る場合は影響度が極度に小さいと判断して削除する。また、初期段階の学習時に誘導点が過度に削除されるのを防ぐとともに、ノード追加時に追加したノードがすぐに削除されないようにするために、削除条件に当てはまった回数が一定値を超えた際に削除する条件と、重みの閾値が学習とともに増加する仕組みを導入している。誘導点の削除の疑似コードをAlgorithm.2に示し、イメージをFigure.3に示す。

Algorithm.2 Delete a Self-moving Point

Constant: $DELW$: 誘導点の削除閾値(重み), $DELR$: 誘導点の削除閾値 (半径), $COUNT$: 該当回数閾値

Require: i : 誘導点番号, w_i : 誘導点の重み, r_i : 誘導点の半径, c_i : 削除カウント, φ_i : i番目のパラメータ集合, $delW$: 誘導点の削除閾値(重み), $epoc$: 学習回数

1: $delW = \sin(epoc)$
//sine関数を用いて$delW$を0~$DELW$に増加
2: if $w_i \leq delW$ && $r_i \leq DELR$ then
3: c_i+= 1
4: if $c_i \geq COUNT$ then
5: delete(φ_i)//i番目のパラメータを削除
6: end if
7: end if

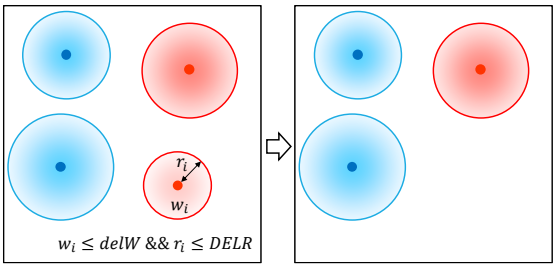

Figure.3 誘導点の削除

3.3　誘導点の結合

誘導点の結合は、類似している2つの誘導点を結合する。これは、類似する誘導点を結合させることで、表現力を落とさずに計算効率を上げることを目的としている。類似度の判断基準として、2点の誘導点パラメータ(座標・半径)の差分が閾値以下であるとともに、重みの符号が等しい場合に、類似性が高いと判断して、2点を結合させる。誘導点の結合の疑似コードをAlgorithm.3に示し、イメージをFigure.4に示す。

Algorithm.3 Merge a Self-moving Point

Constant: MR : 誘導点の結合範囲倍率, MD : 2つの誘導点の半径差分の閾値

Require: i, j, new : 誘導点番号, $\varphi_i, \varphi_j, \varphi_{new}$: 各パラメータ集合, p_i, p_j, p_{new} : 各誘導点の位置, w_i, w_j, w_{new} : 各誘導点の重み, r_i, r_j, r_{new} : 各誘導点の半径

1: if $(\|r_i - r_j\| < MD)$ then
2:　if $(w_i, w_j \geq 0 \,||\, w_i, w_j < 0)$ then
3:　　if $\left(\|p_i - p_j\|_2 < r_i MR \;\&\&\; \|p_i - p_j\|_2 < r_j MR\right)$ then
4:　　add(φ_{new}) // パラメータ集合newを追加
5:　　$p_{new} \leftarrow$neighborhood (p_i, p_j) //重みが大きい点の座標近辺に配置
6:　　$w_{new} \leftarrow (w_i + w_j)$
7:　　$r_{new} \leftarrow \max(r_i, r_j)$
8:　　delete(φ_i, φ_j) // 元の2点を削除
9:　end if
10:　end if
11: end if

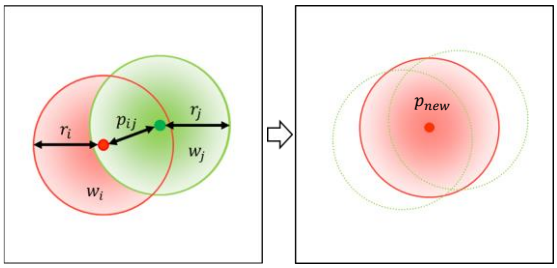

Figure.4　誘導点の結合

4.　実験および検討

本研究では、画像分類タスクにおける分類精度と計算速度を比較する。実験はMNISTデータセットで行う。実験では、従来研究と提案手法の精度比較、提案手法の各要素の精度比較を行う。初期誘導点はすべて16個に設定して実験を行った。

従来手法と提案手法の比較実験では、従来手法(SMP)、従来手法の誘導点を独立して学習できるようにしたもの(SMP(N))、提案手法(SOICC)の3つの手法で比較実験を行う。3つの手法の実験結果をTable.1に示す。

Table.1 従来研究と提案手法の比較

	TraAcc	TesAcc	Time(s)	Point
SMP	0.9432	0.9436	2892	1152
SMP (N)	0.9498	0.9513	3349	1152
SOICC	0.9513	0.9529	3273	1131

Table.1の1列目が学習精度、2列目がテスト精度、3列目が学習時間、4列目が誘導点の数となっている。この実験結果から、SOICCの精度が最も高くなったため、誘導点数を学習させることで精度を向上させられることが示された。しかし、学習時間はSMPよりも遅いため、SOICCは学習時間と精度のトレードオフの関係にあるといえる。また、SOICCの誘導点数がSMPよりも少ないのにも関わらず学習速度が遅いのは、誘導点の増減アルゴリズムを実行する時間と、学習中に誘導点が大きく増減していることが関係していると考えられる。

提案手法の比較実験では、誘導点の増減に用いる「誘導点の追加」、「誘導点の削除」、「誘導点の結合」のうちの2つを組み合わせて、「追加+削除」、「削除+結合」、「結合+追加」の3つの手法で比較実験を行う。3つの手法の実験結果をTable.2に示す。

Table.2 提案手法の比較

	TraAcc	TesAcc	Time(s)	Point
追加+削除	0.9474	0.9492	3371	1193
削除+結合	0.9532	0.9543	3184	1055
結合+追加	0.9530	0.9545	3453	1182

Table.2の結果より、「誘導点の削除」は精度を落とさずに学習時間を短くしていることがわかる。また、「結合+追加」の学習時間が最も長くなっていることからも、誘導点の削除が学習時間を短くすることに対して有効であると言える。「結合と追加」のほうが「追加+削除」よりも精度が高いことから、類似性の高い誘導点を結合することが、精度向上に有効であるといえる。

5. まとめ

本研究では、誘導点数を各レイヤーのフィルタごとに設定することで、学習精度を向上させることができるという仮説のもと、誘導点数を学習によって増減させる手法を提案した。実験結果より、従来手法よりも精度を向上させることができることを示した。また、誘導点の増減に用いる3つのアルゴリズムのうち、誘導点の削除が精度を落とすことなく学習時間を減らせることを示した。

しかし、初期の誘導点が極端に少なかった場合、誘導点の削除が精度の低下を招く可能性があるため、誘導点の初期値を大きく減らして実験を行い、適切に動作しているかを確認する必要がある。また、誘導点の追加と誘導点の結合のアルゴリズムが精度の向上にどれだけ影響を与えているかを、実験をとおして明確に示すことができなかった。そのため、今後は実験環境やパラメータを変更して実験を行い、誘導点の追加と誘導点の結合のアルゴリズムがどのように精度向上にかかわっているのかを明確にしたい。

また、誘導点の増減を実現させるうえでパラメータ数が増加してしまい、パラメータ調整が煩雑になってしまった。そのため、精度を落とすことなく誘導点の増減に必要なパラメータ数を減らす手法についても調査し、実験を行いたい。

参考文献

[1] Vitor Guizilini and Fabio Ramos, “Continuous Convolutional Neural Networks for Image Classification”, OpenReview (2018).
[2] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky and Raquel Urtasun, “Deep Parametric Continuous Convolutional Neural Networks”, arXiv preprint arXiv:2101.06742 (2018).
[3] Sanghyeon Kim and Eunbyung Park, “SMPConv: Self-moving Point Representations for Continuous Convolution”, arXiv preprint arXiv:2304.02330 (2023).