KYOTO UNIVERSITY

CVPR2019読み会
「カーネル畳み込みニューラルネットによる畳み込み操作の非線形化」
Kervolutional Neural Networks

Cheng Wang, Jianfei Yang, Lihoa Xie, Junsong Yuan
読み手：Hisashi Kashima (KU)

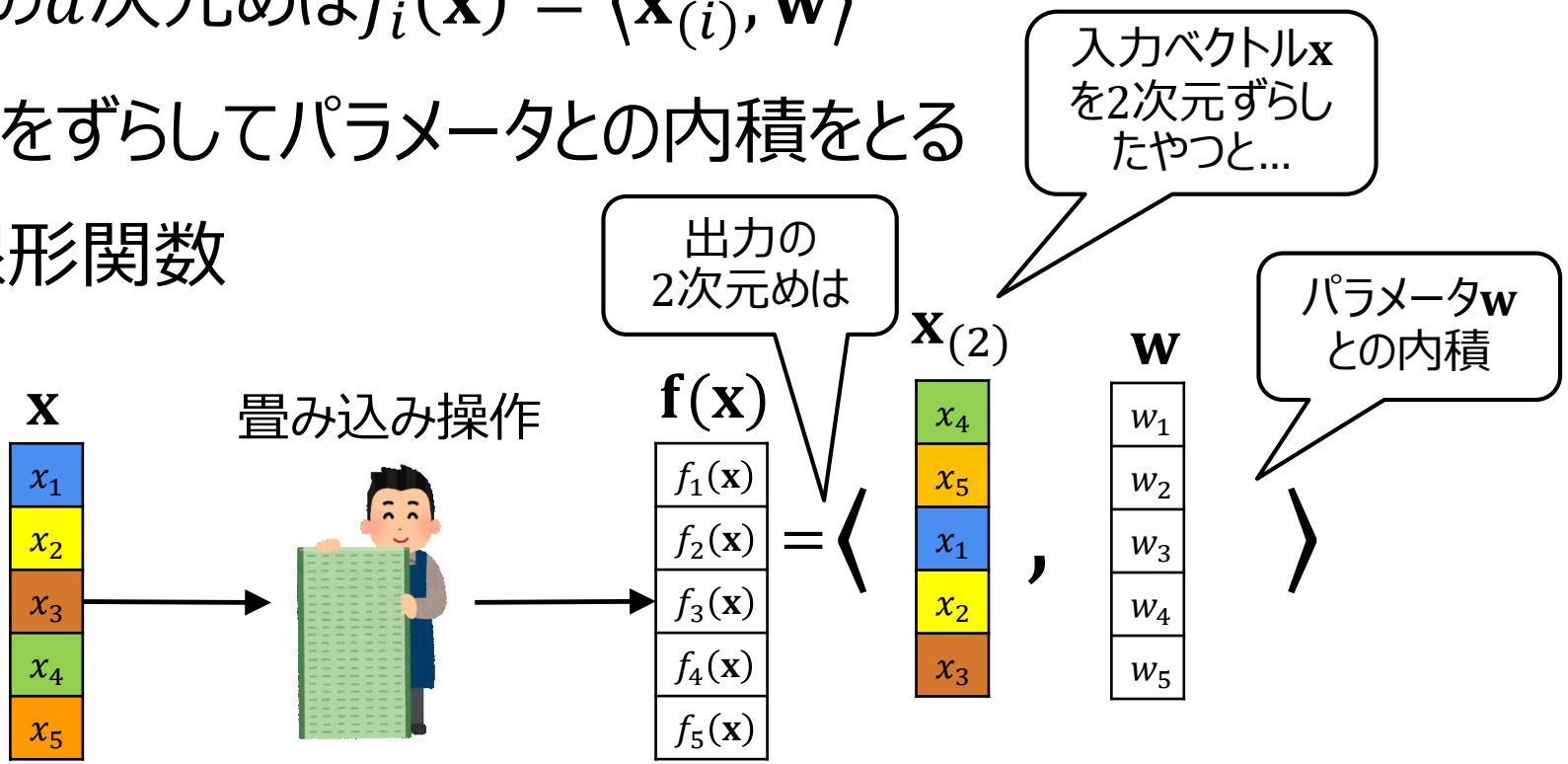
DEPARTMENT OF INTELLIGENCE SCIENCE AND TECHNOLOGY

論文の概要：
畳み込み操作をカーネル関数で非線形化した

- 背景：畳み込みニューラルネットワーク（CNN）はその高性能からあらゆる領域で用いられている
  - 畳み込み操作は線形
- 貢献：CNN＋カーネル＝カーネル畳み込み操作（Kervolution）
  - 畳み込み操作をカーネル関数で非線形化する
  - これで畳み込みを置き換えると性能が上がった
  - 他に非線形性のないネットワークでも、これひとつでかなりいく

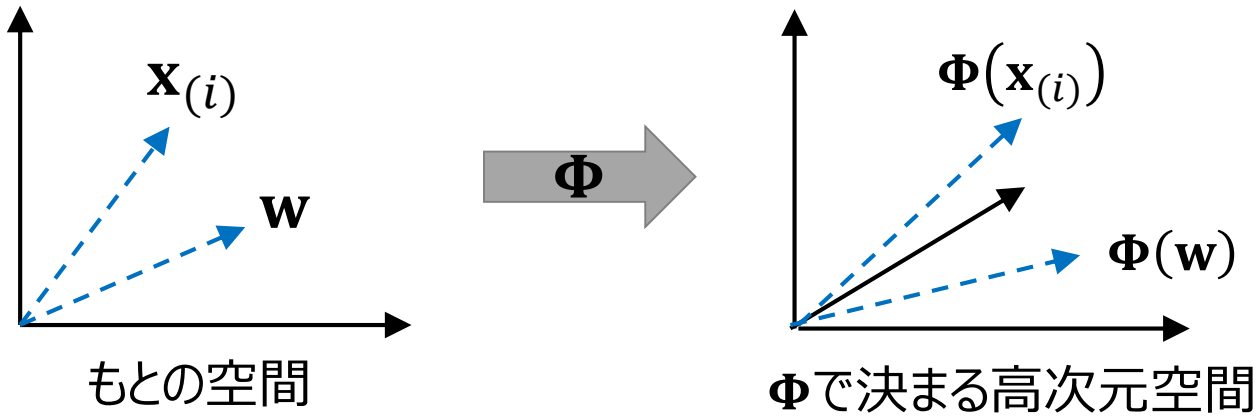
畳み込み操作：畳み込み操作は線形関数

- 畳み込み操作：$\mathbf{f}(\mathbf{x}) = \mathbf{x} \oplus \mathbf{w}$
  - $\mathbf{x}$：入力ベクトル； $\mathbf{w}$：パラメータベクトル
- 出力$\mathbf{f}(\mathbf{x})$のd次元めは$f_i(\mathbf{x}) = \langle \mathbf{x}_{(i)}, \mathbf{w} \rangle$
  - $\mathbf{x}$の次元をずらしてパラメータとの内積をとる
  - これは線形関数



【提案手法】カーネル畳み込み：畳み込み操作の内積部分を非線形化

- $\mathbf{f}(\mathbf{x}) = \mathbf{x} \oplus \mathbf{w}$ のd次元めは $f_i(\mathbf{x}) = \langle \mathbf{x}_{(i)}, \mathbf{w} \rangle$ （線形関数）
- これを 非線形化する： $\mathbf{g}(\mathbf{x}) = \mathbf{x} \otimes \mathbf{w}$ （Φで決まる）
  - （シフトした）入力$\mathbf{x}_{(i)}$を高次元空間に飛ばす $\Phi(\mathbf{x}_{(i)})$ 一般化
  - パラメータ$\mathbf{w}$も同様に飛ばす $\Phi(\mathbf{w})$
  - 飛んだ先で両者の内積をとる $g_i(\mathbf{x}) = \langle \Phi(\mathbf{x}_{(i)}), \Phi(\mathbf{w}) \rangle$



カーネル関数：
高次元空間での内積を効率的なカーネル関数で置き換え

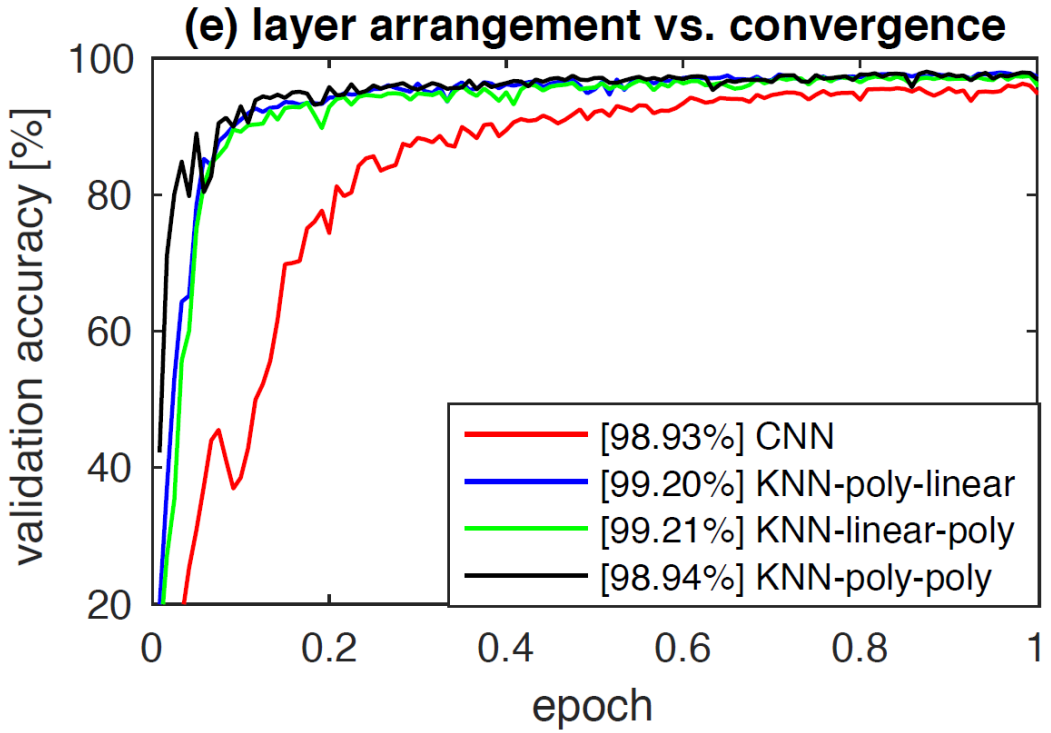
- 内積 $g_i(\mathbf{x}) = \langle \mathbf{\Phi}(\mathbf{x}_{(i)}), \mathbf{\Phi}(\mathbf{w}) \rangle$ をカーネル関数で置き換える：

$$g_i(\mathbf{x}) = \langle \mathbf{\Phi}(\mathbf{x}_{(i)}), \mathbf{\Phi}(\mathbf{w}) \rangle = \kappa(\mathbf{x}_{(i)}, \mathbf{w})$$

- カーネル関数のバリエーション：
  - 多項式カーネル：$\kappa(\mathbf{x}_{(i)}, \mathbf{w}) = (\langle \mathbf{x}_{(i)}, \mathbf{w} \rangle + C)^d$
    - d個までの特徴量の組み合わせを考慮できる
  - ガウシアンカーネル： $\kappa(\mathbf{x}_{(i)}, \mathbf{w}) = \exp(-\gamma \|\mathbf{x}_{(i)} - \mathbf{w}\|)$
  - あと、カーネルではないが単に距離$\|\mathbf{x}_{(i)} - \mathbf{w}\|$なども利用

注目すべき実験結果①：
カーネル畳み込みを用いると収束が早い

- LeNet-5の2つの畳み込み層をカー畳層で置き換えてみる
  - データセット：MNIST
- カーネル畳み込み層を使ったほうが収束が早い



注目すべき実験結果②：畳み込み層の置き換えで性能アップ⇈

- 画像認識（CIFER-10/100, ImageNet）で検証
- 定番ネットワーク(GoogLeNet, ResNet, DenseNet)の1層目をカーネル畳み込み層で置き換えると性能向上

Network	CIFAR-10	CIFAR-100
CNN [26]	13.63	44.74
KNN	10.85	37.12

Table 2. Validation error (%) of ResNets on CIFAR-10 and CIFAR-100 without data augmentation.

Architecture	CIFAR-10+		CIFAR-100+	
	CNN	KNN	CNN	KNN
GoogLeNet [13]	13.37	5.16	26.65	20.84
ResNet [20]	6.43	4.69	27.22	22.49
DenseNet [25]	5.24	5.08	24.42	24.92

Table 3. Validation error (%) on CIFAR-10 and CIFAR-100 on different architectures with data augmentation.

注目すべき実験結果③：
カーネル畳み込みが入れる非線形はかなり強力

- CNNにおいて、モデルに非線形性をもたらす部分を除く
  - ReLU を 恒等写像（線形）に
  - 最大プーリング を 平均値プーリング（線形）に

  ⇒ MNISTで 98.0% → 92.2% と大幅悪化
- そこに、非線形要素としてカー畳をいれると、これだけで 99.11%
  - 2つの畳み込み層をカーネル畳み込み層にする
- NNに入れる非線形性はこれで十分である可能性も

論文の概要：
畳み込み操作をカーネル関数で非線形化したらよかった

- CNNにおける畳み込み操作を、カーネル化することで、収束が早く、精度が向上した
- 考察：この論文での「カーネル化」は従来のカーネル化よりもちょっと弱い
  - 通常のカーネル法では$\mathbf{\Phi}$で行った先での任意の線形関数（が元の食空間では非線形にみえる）を考えるが、この論文ではとれる関数に制約がかかる
    - 本来の形：$g_i(\mathbf{x}) = \langle \mathbf{\Phi}(\mathbf{x}_{(i)}), \mathbf{w} \rangle$
    - この論文：$g_i(\mathbf{x}) = \langle \mathbf{\Phi}(\mathbf{x}_{(i)}), \mathbf{\Phi}(\mathbf{w}) \rangle$ （自由度低い）