

第132回 人工知能基本問題研究会 2025年3月19日

非対称ATTENTIONを用いた集合埋め込みと ヘテロジニアス集合マッチング・検索への応用



和歌山大学・大学院システム工学研究科
理研・革新知能統合センター
八谷 大岳

※本研究はJSPS科研費JP23K11218の助成を受けたものです。

集合データとは

2

Y. Saito et al., ECCV2020

□ 重複せず順序を持たない複数の要素が組み合わさったデータ

□ 例 :

- **ファッションや家具コーデ** : 調和する複数のアイテム (画像、商品名、属性など) の集合

夏の30代ママコーデ

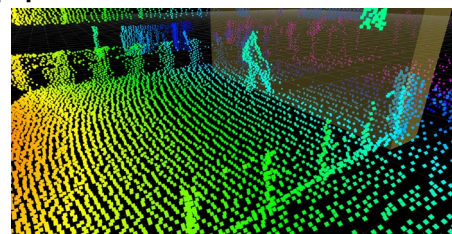
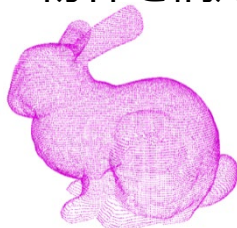
<https://blog.smasell.jp/archives/5347>



- **人のグループ** : 共通の趣味や目的を持つ人 (属性、画像など) の集合



- **点群データ** : 物体を構成する3次元の点の集合



<https://tech-deliberate-jiro.com/3dmodel/>

<https://www.blickfeld.com/blog/on-device-motion-detection/>

定式化

3

- 集合 \mathcal{X} : ベクトル $\mathbf{x}_i \in \mathbb{R}^{1 \times D}$ を要素に持つ集合

$$\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N^{\mathcal{X}}}\} \quad N^{\mathcal{X}}: \text{要素数}$$

- 順列生成関数 Π : 集合 \mathcal{X} の全ての順列の集合を生成

- 例) $\mathcal{X} = \{a, b, c\}$ の場合

➡ $\Pi(\mathcal{X}) = \{a, b, c, \{a, c, b\}, \{b, a, c\}, \{b, c, a\}, \{c, a, b\}, \{c, b, a\}\}$

- **Permutation invariance (順不変)** : $f(\mathcal{X}) = f(\pi\mathcal{X})$

$$\pi\mathcal{X} \in \Pi(\mathcal{X}) \quad \pi : \text{順列を変更する作用素 (順列操作作用素)}$$

- **Permutation equivalence (順同変)** : $f(\pi\mathcal{X}) = \pi f(\mathcal{X})$

$$f(\mathcal{X}) = \{\|\mathbf{x}\|^2; \mathbf{x} \in \mathcal{X}\}$$
$$f(\pi\mathcal{X}) = f(\{b, c, a\}) = \{b^2, c^2, a^2\}$$
$$\pi f(\mathcal{X}) = \pi f(\{a, b, c\}) = \pi\{a^2, b^2, c^2\} = \{b^2, c^2, a^2\}$$

- 目的 : **順不変性**を持ち**適応的**な関数 $f(\mathcal{X})$ を作りたい

内容：

4

- 集合データとは
- 様々な順不変関数
- 順不変関数のまとめ
- 集合マッチングへの応用
- 集合検索への発展とまとめ

単純な順不変関数 : sum pooling

5

- sum poolingを用いた順不変性を満たす関数

$$f(\mathcal{X}) = \sum_{i=1}^{N_{\mathcal{X}}} x_i$$

- $\mathcal{X} = \{4, 1, -3\}$ の場合の全ての順列 $\Pi(\mathcal{X})$:

$$\Pi(\mathcal{X}) = \{\{4, 1, -3\}, \{4, -3, 1\}, \{1, 4, -3\}, \{1, -3, 4\}, \{-3, 4, 1\}, \{-3, 1, 4\}\}$$

$$f(\{4, 1, -3\}) = 4 + 1 - 3 = 2$$

$$f(\{1, -3, 4\}) = 1 - 3 + 4 = 2$$

$$f(\{4, -3, 1\}) = 4 - 3 + 1 = 2$$

$$f(\{-3, 4, 1\}) = -3 + 4 + 1 = 2$$

$$f(\{1, 4, -3\}) = 1 + 4 - 3 = 2$$

$$f(\{-3, 1, 4\}) = -3 + 1 + 4 = 2$$



$$f(\mathcal{X}) = f(\pi\mathcal{X})$$

$$\pi\mathcal{X} \in \Pi(\mathcal{X})$$

Deep sets

☹️ 単純なsum poolingのみでは適応的な処理は不可

□ **Deep sets**: 以下の定理を提案

Theorem 2 A function $f(X)$ operating on a set X having elements from a countable universe, is a valid set function, i.e., **invariant** to the permutation of instances in X , iff it can be decomposed in the form $\rho\left(\sum_{x \in X} \phi(x)\right)$, for suitable transformations ϕ and ρ .

□ 順不変な関数 $f(X)$ は必ず以下の形に分解できる

$$f(X) = \rho\left(\sum_{x \in X} \phi(x)\right)$$

$\phi(x)$: 要素に適用される適切な変換関数

$\rho(\cdot)$: 最終的な出力を決める適切な変換関数

□ $\phi(\cdot)$ と $\rho(\cdot)$ を、MLPなどの学習可能なモデルで表現することにより、
順不変かつ適応的な関数 $f(X)$ を実現

☹️ $\phi(\cdot)$ は、各要素 x を独立に処理するため、集合内の要素の共起性に基づく
変換が不可

Set TransformerとMAB

9

J. Lee et al., ICML2019

- 集合 \mathcal{X} 、 \mathcal{Y} : ベクトル \mathbf{x} 、 $\mathbf{y} \in \mathbb{R}^{1 \times D}$ を要素に持つ集合
 $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N^X}\}$ $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_{N^Y}\}$ N^X 、 N^Y : アイテム数
- 計算の便宜上、集合 \mathcal{X} 、 \mathcal{Y} を行列 X 、 Y で表現

$$X = [\mathbf{x}_1; \mathbf{x}_2; \dots; \mathbf{x}_{N^X}] \in \mathbb{R}^{N^X \times D} \quad Y = [\mathbf{y}_1; \mathbf{y}_2; \dots; \mathbf{y}_{N^Y}] \in \mathbb{R}^{N^Y \times D}$$

- Multi-head attention block (MAB) :**

- Multi-head attention:

$$\text{attention}(Q, K, V) = \text{Softmax}\left(\frac{1}{\sqrt{D}} QK^T\right)V$$

$$Q'_h = \text{Attention}(QW_h^Q, KW_h^K, VW_h^V)$$

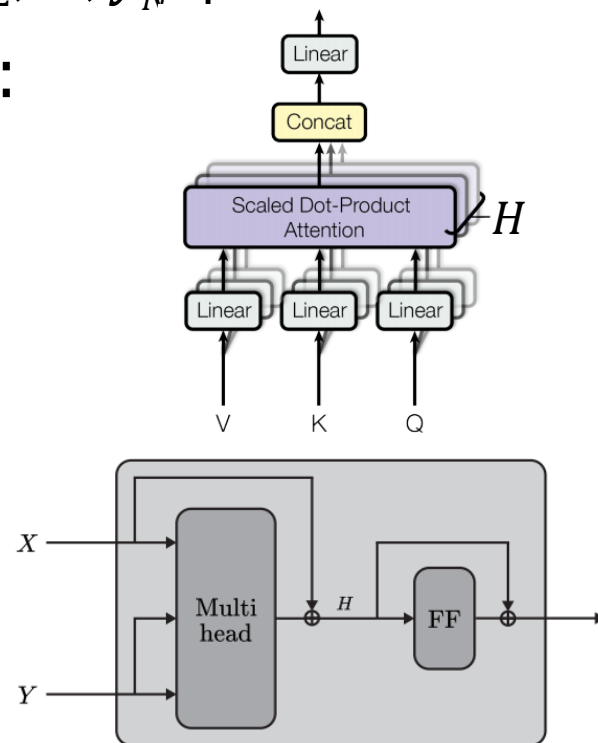
$$\text{MultiHead}(Q, K, V) = \text{Concat}(Q'_1, Q'_2, \dots, Q'_H)W^H$$

- MAB:

$$\text{MAB}(X, Y) = \text{LayerNorm}(H + \text{rFF}(H))$$

$$H = \text{LayerNorm}(X + \text{MultiHead}(X, Y, Y))$$

rFF : 要素独立な (次元方向の) 全結合層



Attentionの性質とSAB

10

- Attention: 順同変と要素単位の順不変を同時に満たす

$$Q' = \text{Attention}(Q, K, V) = \left(\begin{array}{c} \boxed{q_1} \\ \boxed{q_2} \end{array} \right) \left(\begin{array}{c} \boxed{k_1} \\ \boxed{k_2} \\ \boxed{k_3} \end{array} \right) \left(\begin{array}{c} \boxed{v_1} \\ \boxed{v_2} \\ \boxed{v_3} \end{array} \right)$$

$$= \left(\begin{array}{c} q_1 k_1^T v_1 + q_1 k_2^T v_2 + q_1 k_3^T v_3 \\ q_2 k_1^T v_1 + q_2 k_2^T v_2 + q_2 k_3^T v_3 \end{array} \right)$$

※ $\frac{1}{\sqrt{D}}$ 、softmaxは省略

順同変: $f(\pi Q) = \pi f(Q)$
出力 Q' と Q'' の順列は入力 Q の順列と同じ

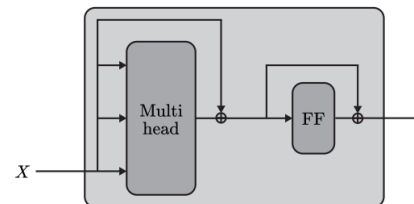
$$Q'' = \text{Attention}(Q, K, V) = \left(\begin{array}{c} \boxed{q_2} \\ \boxed{q_1} \end{array} \right) \left(\begin{array}{c} \boxed{k_3} \\ \boxed{k_2} \\ \boxed{k_1} \end{array} \right) \left(\begin{array}{c} \boxed{v_3} \\ \boxed{v_2} \\ \boxed{v_1} \end{array} \right)$$

$$= \left(\begin{array}{c} q_2 k_3^T v_3 + q_2 k_2^T v_2 + q_2 k_1^T v_1 \\ q_1 k_3^T v_3 + q_1 k_2^T v_2 + q_1 k_1^T v_1 \end{array} \right)$$

順不変: $f(X) = f(\pi X)$
出力 Q' と Q'' の各要素の値は、
入力 K の順列に依存しない

- Set Attention Block (SAB) : MABをself-attentionで処理

$$\text{SAB}(X) = \text{MAB}(X, X)$$

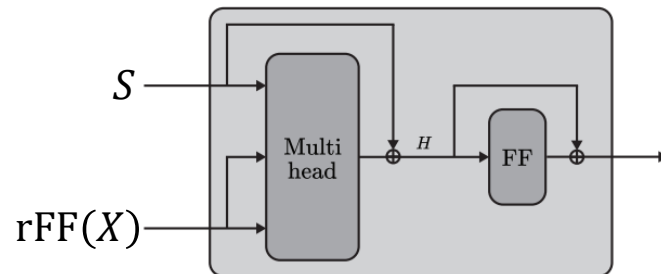


Pooling MA (PMA)

11

- Q を学習可能な行列 $S \in \mathbb{R}^{K \times D}$ に置き換え、順不変性を実現

$$\text{PMA}(X) = \text{MAB}(S, \text{rFF}(X))$$



- $K = 1$ の場合の例 :

$$\begin{bmatrix} S' \\ \text{---} \end{bmatrix} = \text{softmax} \left(\frac{1}{\sqrt{D}} \begin{bmatrix} S \\ \text{---} \end{bmatrix} \begin{bmatrix} X^T \\ \text{---} \end{bmatrix} \right) \begin{bmatrix} X \\ \text{---} \end{bmatrix}$$

- S と X の関係に基づく重み係数を用いて、 X の線形結合によりpooling
- 出力 S' は、入力 X の順列に依存しない

$$\begin{bmatrix} S'' \\ \text{---} \end{bmatrix} = \text{softmax} \left(\frac{1}{\sqrt{D}} \begin{bmatrix} S \\ \text{---} \end{bmatrix} \begin{bmatrix} X^T \\ \text{---} \end{bmatrix} \right) \begin{bmatrix} X \\ \text{---} \end{bmatrix}$$

$S' = S''$

内容：

12

- 集合データとは
- 様々な順不変関数
- 順不変関数のまとめ
- 集合マッチングへの応用
- 集合検索への発展とまとめ

順不変関数のまとめ

$\Pi(\mathcal{X})$: 集合 \mathcal{X} の全順列の集合 (順列生成関数)

π : 順列を変更する作用素 (順列操作作用素)

13

- **Deep sets** : $f(\mathcal{X}) = \rho(\sum_{x \in \mathcal{X}} \phi(x))$ M. Zaheer, NeurIPS 2017
 - Pooling&順不変性 : sum-pooling
 - $\phi(\cdot)$: 入力は要素 x

- **PointNet** : $f(\mathcal{X}) = \rho\left(\max_{x \in \mathcal{X}} \phi(x)\right)$ C. R. Qi et al., CVPR2017
 - Pooling&順不変性 : max-pooling
 - $\phi(\cdot)$: 入力は要素 x

- **Janossy pooling** : $f(\mathcal{X}) = \rho\left(\frac{1}{|\Pi(\mathcal{X})|} \sum_{\pi \mathcal{X} \in \Pi(\mathcal{X})} \phi(\pi \mathcal{X})\right)$
 - Pooling&順不変性 : sum-pooling R. L. Murphy et al., ICLR2019
 - $\phi(\cdot)$: 入力は集合 $\pi \mathcal{X}$

順不変関数のまとめ 2

14

- **SetNet (NetVLAD)** : $f(\mathcal{X}) = \rho(\sum_{\mathbf{x} \in \mathcal{X}} \phi_2(\mathbf{x})\phi_1(\mathbf{x}))$
 - Pooling&順不変性 : 線形結合
 - $\phi_1(\mathbf{x}) = \{\mathbf{x} - \mathbf{c}_k; k = 1, 2, \dots, C\}$: クラスタ-との差 Y. Zhong et al., ECCV2018
 - $\phi_2(\mathbf{x}) = \text{softmax}(\{\mathbf{x}\mathbf{w}_k + b_k; k = 1, 2, \dots, C\})$: 重み係数
 - ρ : クラスタ- k に関する正規化和

- **PoolingMA** : $f(\mathcal{X}) = \rho(\sum_{\mathbf{x} \in \mathcal{X}} \phi_2(\mathbf{x})\phi_1(\mathbf{x}))$
 - Pooling&順不変性 : 線形結合
 - $\phi_1 = \mathbf{x}W^V$
 - $\phi_2 = \text{softmax}(\{(\mathbf{s}W^Q)(\mathbf{x}W^K)^T\}; \mathbf{x} \in \mathcal{X})$: 重み係数 J. Lee et al., ICML2019

- **DuMLP-pin**: $f(\mathcal{X}) = \rho(\sum_{\mathbf{x} \in \mathcal{X}} \phi_2(\mathbf{x})\phi_1(\mathbf{x}))$
 - Pooling&順不変性 : 線形結合 J. Fei et al., AAAI2022
 - $\phi_1(\cdot)$ 、 $\phi_2(\cdot)$: 異なるMLP (片方が重み係数の役割)

内容：

16

- 集合データとは
- 様々な順不変関数
- 順不変関数のまとめ
- 集合マッチングへの応用
- 集合検索への発展とまとめ

集合マッチングとは

- クエリ集合 $X \in \mathbb{R}^{N^X \times D}$ と調和する集合 $Y^* \in \mathbb{R}^{N^Y \times D}$ をギャラリー \mathcal{G} から選択

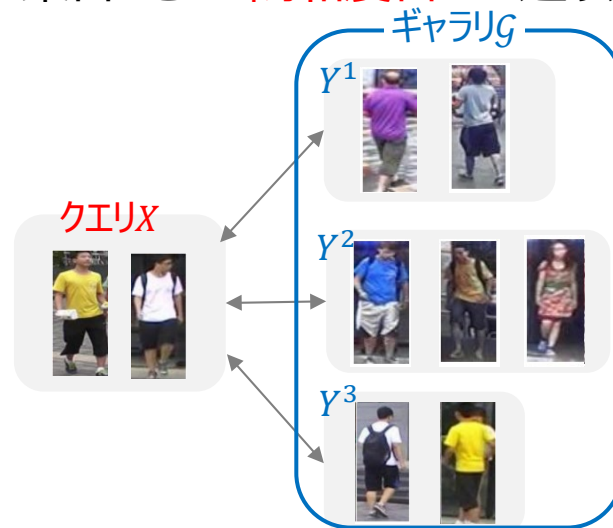
$$Y^* = \operatorname{argmax}_{Y \in \mathcal{G}} f(X, Y)$$

$$\mathcal{G} = \{Y^1, Y^2, \dots, Y^{N^g}\}$$

$f(X, Y) \in [0, 1]$: 集合 X と Y の調和度合いの返すスコア関数



ファッションコーデ推薦



Group ReID

- スコア関数 $f(X, Y)$ の要件 :

- 集合間の交換性 : $f(X, Y) = f(Y, X)$
- 要素間の順不変性 : $f(X, Y) = f(\pi X, \pi Y)$



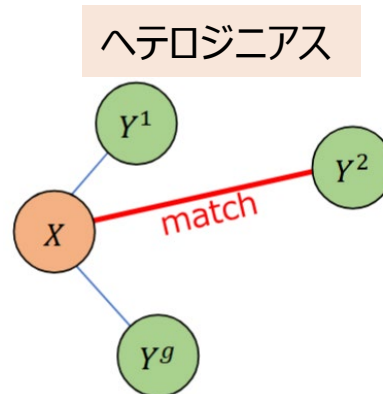
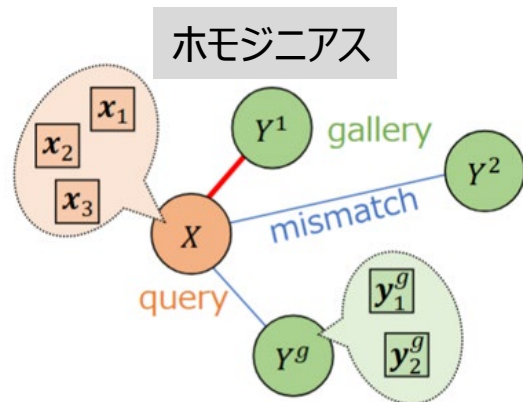
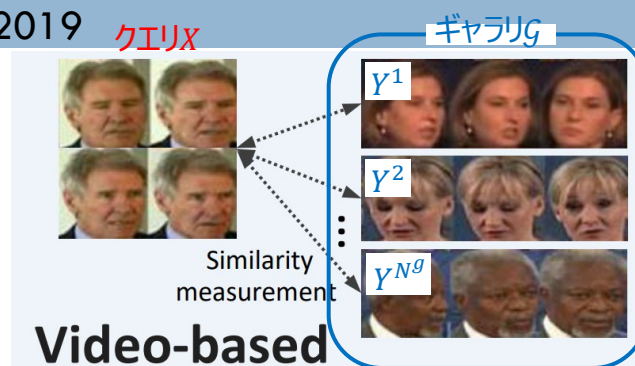
$$f(X, Y) = f(\pi Y, \pi X)$$

集合マッチングの種類

18

H. Hachiya & Y. Saito, Neurocomputing2024 X. Liu et al., ICCV2019

- ホモジニアス集合マッチング：
 - 同一物体や人物を含む集合同士のマッチング問題
 - Group Re-IDや動画を用いた顔・物体認識など
 - 特徴空間上で類似した集合同士がマッチング



- ヘテロジニアス集合マッチング：
 - 異なる種類の物体の集合同士のマッチング問題
 - ファッションや家具のコーデ推薦など
 - 特徴空間上で離れた集合同士でも調和

➡ 調和する離れたベクトルを近づける変換が要

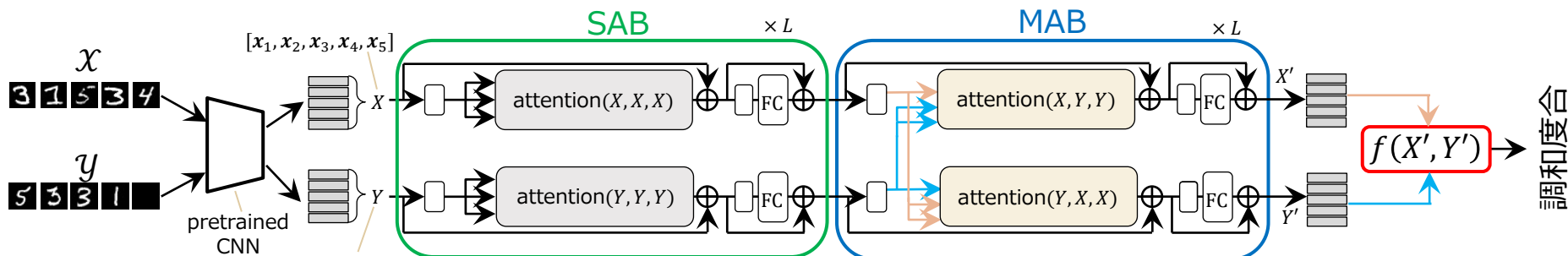


Attentionベース集合マッチング

19

Saito, Nakamura, Hachiya, Fukumizu, ECCV2020

- SABとMABを繰り返す、集合 x と y の要素ベクトルを変換



☹️ X' と Y' は **順同変** なので、スコア関数 $f(X, Y)$ にて **順不変** を満たす必要

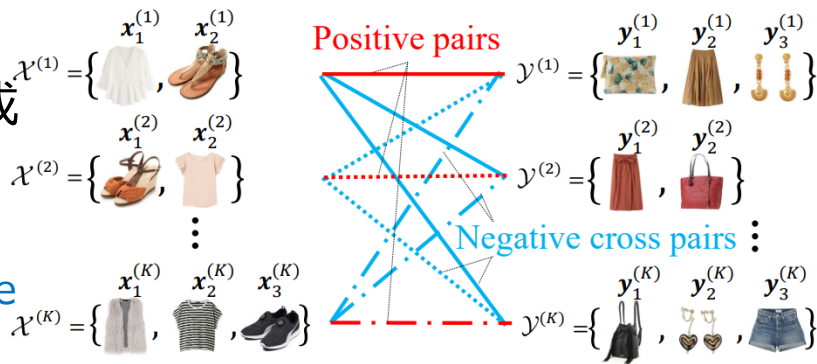
- **Cross similarity score (CSS)** : 全ペア間の内積

$$f(x, y) = \frac{1}{N^X N^Y} \sum_{x \in X} \sum_{y \in Y} \text{ReLU} \left(\frac{(xW)(yW)^T}{\sqrt{D}} \right) \quad W \in \mathbb{R}^{D \times D}$$

集合交換性と要素順不変性を満たす

- **学習 : K-pair-set loss**

- K 個の集合をランダムに選択しミニバッチを作成
 - 各集合をランダムに $x^{(k)}$ と $y^{(k)}$ に分割
 - $x^{(k)}$ と $y^{(k)}$ を **positive**
 - $x^{(k)}$ と $y^{(k' \neq k)}$ および $x^{(k' \neq k)}$ と $y^{(k)}$ を **negative**
- 交差エントロピー最小化によりモデルを学習



Attentionベース集合間マッチングの課題

20

H. Hachiya & Y. Saito, Neurocomputing2024

□ スコア計算 $f(X, Y)$ における勾配消失

• CSSの場合 :

$$f(X, Y) = \frac{1}{N^X N^Y} \sum_{x \in X} \sum_{y \in Y} \text{ReLU} \left(\frac{(xW)(yW)^T}{\sqrt{D}} \right)$$

内積が負の要素に対する勾配が消失

• max pooling+内積の場合 :

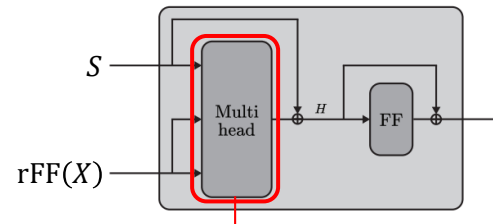
$$f(X, Y) = (\max X)(\max Y)^T \quad \max X \in \mathbb{R}^{1 \times D}$$

最大以外の要素に対する勾配が消失

• PMA+内積の場合 :

$$f(X, Y) = (\text{PMA}(X))(\text{PMA}(Y))^T$$

$$\text{PMA}(X) = \text{MAB}(S, \text{rFF}(X))$$



極端に類似度が高い要素があると、Softmaxにより勾配が消失

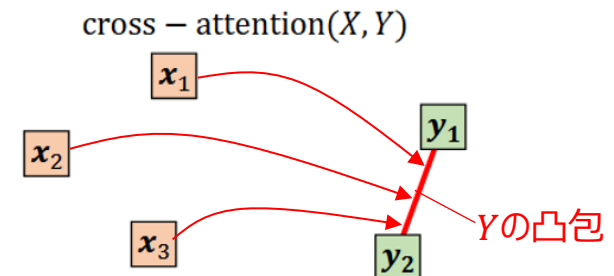
□ Attention : 要素数が少ない場合、変換は限定的

□ Q を V の凸包上での変換

$$\mathbf{x}'_1 = \text{attention}(\mathbf{x}_1, Y, Y) = w_1 \mathbf{y}_1 + w_2 \mathbf{y}_2 + \dots + w_{N_Y} \mathbf{y}_{N_Y}$$

$$w_1 + w_2 + \dots + w_{N_Y} = 1$$

$$w_i \geq 0$$



■ V の要素が2個しかない場合は、線分上に移動

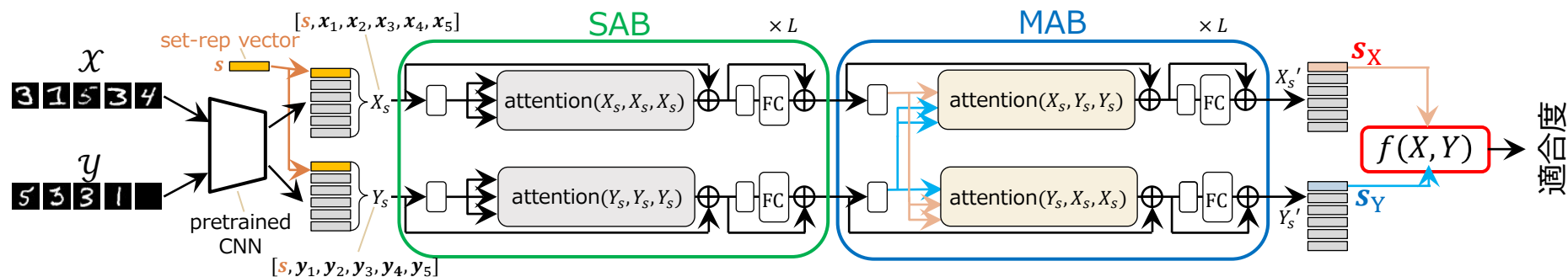
提案法：集合代表ベクトル

21

H. Hachiya & Y. Saito, Neurocomputing2024

- 学習可能ベクトル $\mathbf{s} \in \mathbb{R}^{1 \times D}$ を各集合に要素として追加

$$X_s = [\mathbf{s}; X], Y_s = [\mathbf{s}; Y]$$



- SABとMABを介して変換し、集合を表すベクトル \mathbf{s}_X と \mathbf{s}_Y を獲得

- スコア計算：集合代表ベクトルの内積 $f(X, Y) = \mathbf{s}_X \mathbf{s}_Y^T$

➡ 単純な内積計算のみなので、勾配消失を回避

提案法：非対称Attentionによる集合変換

22

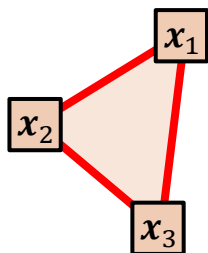
H. Hachiya & Y. Saito, Neurocomputing2024

□ Bi-PMA: Bi-directional PMA

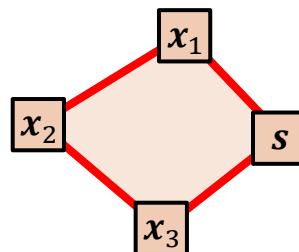
$$X_s' = \text{attention}(X_s, X_s, X_s)$$

- X と s 双方向のattentionにより、広い凸包上で変換

attention(X, X, X)



attention(X_s, X_s, X_s)

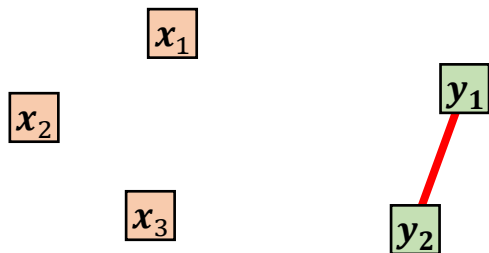


□ Pivot-cross attention

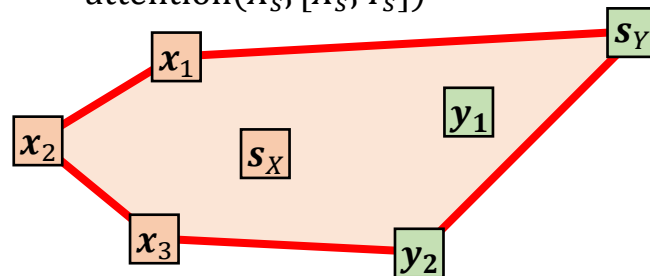
$$X_s' = \text{attention}(X_s, [X_s; Y_s])$$

- KeyとValueに自分自身も入れることにより、より広い凸包上で変換

attention(X, Y, Y)



attention($X_s, [X_s; Y_s]$)



Set-norm & cross-set-norm

23

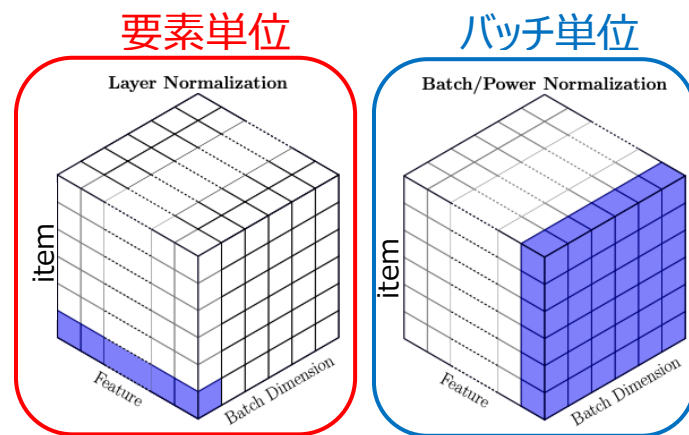
Hachiya & Saito, Neurocomputing2024

L. H. Zhang et al., ICML2022

- Attentionの内積計算：共通の空間で行うべき

$$Q' = \text{Attention}(Q, K, V) = \text{softmax}\left(\left((X_S W^Q)(Y_S W^K)\right)\right)(Y_S W^V)$$

- 既存のNormalizationの正規化対象：



- Attention対象の集合に合わせて正規化

- **Set-norm:** それぞれの集合 X_S 、 Y_S で平均 $\mu \in \mathbb{R}^{1 \times D}$ と標準偏差 $\sigma \in \mathbb{R}^{1 \times D}$ を計算
- **Cross-set-norm:** 集合ペア $[X_S; Y_S]$ 内で共通の平均 μ と標準偏差 σ を計算

$$q'_{id} = \gamma_d \frac{q_{id} - \mu_d}{\sigma_d} + \beta_d$$

集合間マッチングの評価データ

25

Hachiya & Saito, Neurocomputing2024

□ ホモジニアス集合マッチング :

□ Group Re-ID :

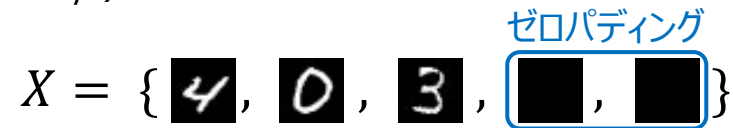
- Market-1501 (<https://docs.cvat.ai/docs/manual/advanced/formats/format-market1501/>)
- ラベル : $l_{XY} = \begin{cases} 1 & \text{if } X \text{と} Y \text{が同じグループ} \\ 0 & \text{otherwise} \end{cases}$
- 例 : $N_X = 3$ の場合



□ ヘテロジニアス集合マッチング :

□ 手書き数字の和の偶奇判別 :

- MNIST (<https://yann.lecun.com/exdb/mnist/>)
- ラベル : $l_{XY} = \begin{cases} 1 & \text{if } X + Y = \text{偶数} \\ 0 & \text{otherwise} \end{cases}$
- 例 : 5以下の数字、最大要素数=5



□ Fashion outfit matching :

- SHIFT15M (<https://github.com/st-tech/zozo-shift15m>)
- ラベル : $l_{XY} = \begin{cases} 1 & \text{if } X \text{と} Y \text{が同じコーデ} \\ 0 & \text{otherwise} \end{cases}$
- 例 : 最大要素数=5の場合



集合間マッチングの精度比較









- 評価指標 CMC= k : 上位 k に正解が入っている割合 (3回の平均と分散)

| | | Attentionの回数 L | ホモジニアス Group Re-ID | | | ヘテロジニアス | | | | |
|--|-------------------------|------------------|-----------------------|-------------------|-------------------|---|-------------------|-------------------------|-------------------|-------------------|
| | | method | CMC=1 | CMC=2 | CMC=3 | Even-total matching ($N=3, N_r=5$) | | Fashion-outfit matching | | |
| | | | | | | CMC=1 | CMC=1 | CMC=1 | CMC=2 | CMC=3 |
| バックボーンの基本形 : SAB × L + MAB × L | maxPool | 1 | 68.8 (2.0) | 88.5 (1.3) | 96.0 (0.7) | 92.0 (2.7) | 59.1 (1.8) | | | |
| | | 3 | 67.2 (21.0) | 88.0 (1.1) | 95.5 (0.6) | 93.9 (0.3) | 66.6 (10.3) | 20.6 (0.6) | 40.1 (0.6) | 60.3 (0.3) |
| | | 5 | 67.8 (2.1) | 87.8 (1.3) | 95.7 (0.5) | 79.0 (24.8) | 65.0 (16.0) | | | |
| 単純なpoolingと内積 | linearProj | 1 | 68.1 (0.5) | 88.7 (1.7) | 96.2 (1.0) | 91.2 (0.4) | 56.1 (6.6) | | | |
| | | 3 | 60.5 (0.5) | 82.4 (5.6) | 92.9 (4.0) | 94.1 (0.7) | 59.1 (2.9) | 21.6 (2.4) | 42.0 (3.9) | 62.2 (3.6) |
| | | 5 | 43.2 (30.4) | 62.2 (36.2) | 75.4 (31.3) | 92.1 (1.9) | 60.5 (13.6) | | | |
| L が大きくなると精度劣化 集合マッチング方式 | PMA | 1 | 67.7 (1.9) | 89.1 (0.5) | 95.8 (0.5) | 91.0 (1.7) | 65.1 (5.6) | | | |
| | | 3 | 63.5 (4.7) | 85.9 (3.0) | 94.7 (1.2) | 94.3 (0.3) | 76.1 (0.8) | 21.0 (1.3) | 40.9 (2.0) | 60.9 (1.3) |
| | | 5 | 50.7 (26.6) | 71.4 (27.8) | 83.5 (20.7) | 78.6 (24.5) | 54.7 (6.9) | | | |
| 動画ベース顔認識方式 | CATSET | 1 | 69.0 (3.2) | 89.1 (2.7) | 96.2 (1.1) | 86.0 (4.2) | 63.7 (2.3) | | | |
| | | 3 | 34.8 (22.8) | 55.3 (24.8) | 71.7 (18.8) | 64.7 (22.7) | 58.2 (7.4) | 20.2 (0.0) | 39.8 (0.0) | 60.1 (0.0) |
| | | 5 | 14.0 (10.4) | 28.1 (19.3) | 45.9 (23.7) | 49.9 (0.4) | 49.9 (0.4) | | | |
| 提案法 : 集合代表ベクトル + Bi-PMA + pivot-cross | CSS | 1 | 67.7 (1.1) | 88.3 (0.7) | 94.8 (1.1) | 91.7 (2.0) | 65.1 (1.6) | | | |
| | | 3 | 24.7 (8.9) | 45.5 (13.0) | 61.7 (15.7) | 89.5 (7.0) | 60.0 (17.9) | 20.2 (0.0) | 39.8 (0.0) | 60.1 (0.0) |
| | | 5 | 17.7 (5.2) | 35.1 (7.9) | 54.2 (9.1) | 50.2 (0.5) | 50.7 (1.7) | | | |
| 提案法 : 集合代表ベクトル + Bi-PMA + pivot-cross | PIFR sparse | 1 | 65.5 (7.5) | 86.3 (5.5) | 94.6 (2.1) | 90.5 (0.6) | 65.7 (1.7) | | | |
| | | 3 | 66.9 (1.4) | 88.2 (1.1) | 95.7 (0.3) | 93.3 (0.9) | 75.4 (6.9) | 33.5 (27.6) | 55.7 (27.6) | 72.4 (21.3) |
| | | 5 | 63.2 (1.2) | 85.3 (1.8) | 94.0 (0.7) | 94.1 (0.2) | 76.3 (9.9) | | | |
| 提案法 : 集合代表ベクトル + Bi-PMA + pivot-cross | PIFR collaborative | 1 | 68.8 (2.1) | 88.7 (0.9) | 96.1 (1.0) | 91.9 (0.6) | 65.6 (0.4) | | | |
| | | 3 | 68.3 (1.9) | 88.9 (1.1) | 95.6 (0.5) | 94.3 (0.3) | 79.1 (6.9) | 32.2 (20.7) | 54.9 (26.2) | 72.2 (20.8) |
| | | 5 | 64.4 (4.1) | 85.4 (4.4) | 94.9 (1.6) | 94.3 (0.5) | 75.8 (9.3) | | | |
| 提案法 : 集合代表ベクトル + Bi-PMA + pivot-cross | bi-PMA + bi-PMA | 1 | 64.4 (1.7) | 86.7 (0.4) | 94.5 (0.8) | 94.9 (0.4) | 83.6 (0.5) | | | |
| | | 3 | 69.1 (3.0) | 88.7 (1.8) | 95.0 (0.9) | 94.6 (0.5) | 87.5 (0.3) | 81.4 (5.3) | 96.3 (1.8) | 99.2 (0.5) |
| | | 5 | 67.3 (1.7) | 88.6 (1.0) | 95.9 (0.4) | 94.6 (0.5) | 75.6 (22.3) | | | |
| 提案法 : 集合代表ベクトル + Bi-PMA + pivot-cross | bi-PMA + pivot-cross | 1 | 65.8 (3.8) | 87.6 (2.8) | 94.9 (1.4) | 95.1 (0.1) | 86.1 (1.2) | | | |
| | | 3 | 65.2 (4.0) | 87.0 (2.6) | 95.3 (1.1) | 94.9 (0.2) | 88.5 (0.6) | 87.9 (1.5) | 98.3 (0.6) | 99.8 (0.1) |
| | | 5 | 68.2 (2.3) | 88.5 (1.8) | 95.8 (1.2) | 95.3 (0.6) | 88.1 (1.0) | | | |

- 提案法 : 階層数 L を増やしても、ヘテロでも高い精度

マッチング結果の例：偶奇判別

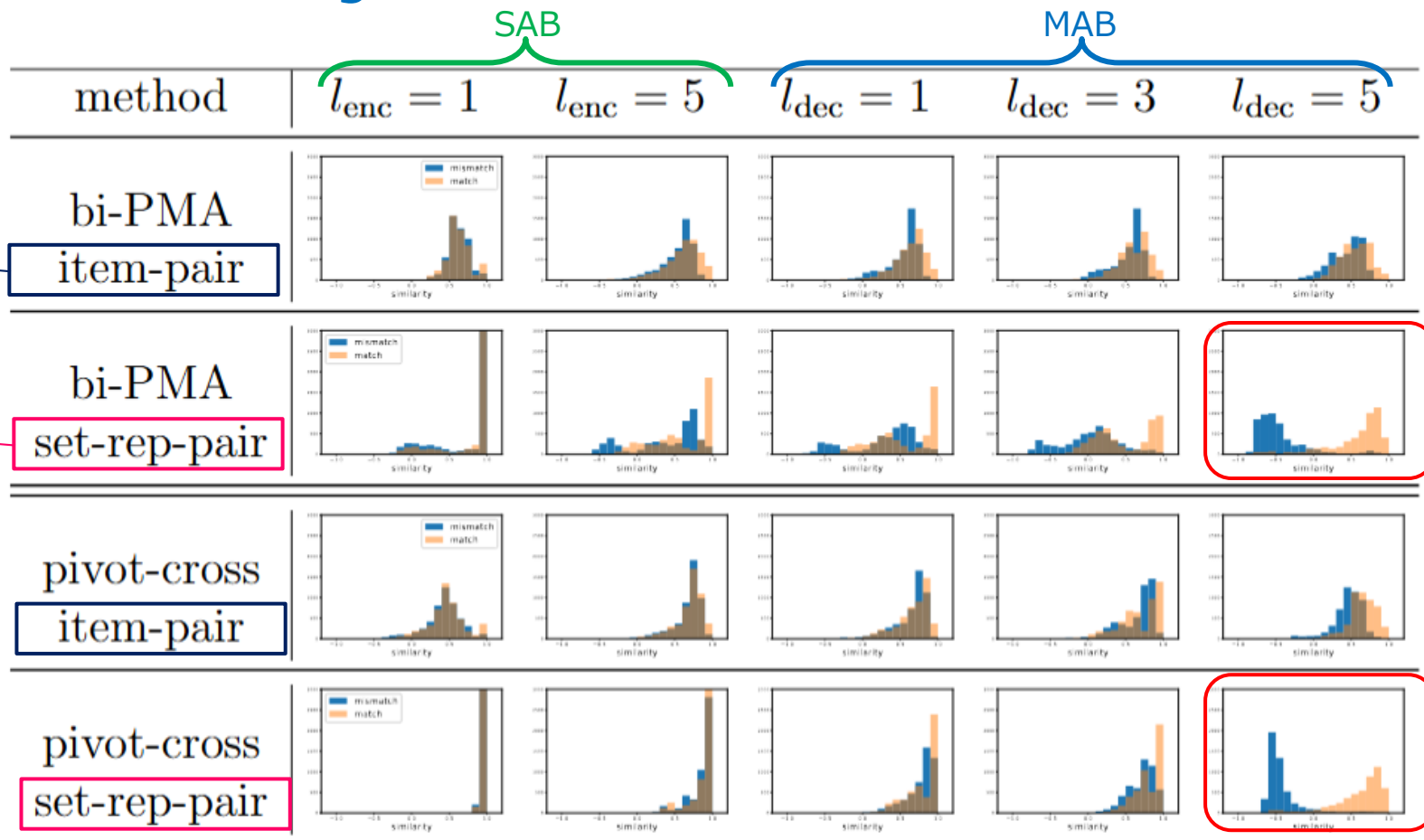
Table 4: Examples of digits images in sets X and Y on even-total matching, and predicted scores using the existing methods, i.e., PMA (Sec. 3.1), CATSET (Sec. 3.2), CSS (Sec. 3.3), and PIFR sparse/collaborative (Sec. 3.4), and the proposed methods, i.e., bi-PMA (Sec. 4.2) and pivot-cross (Sec. 4.3), for the tasks with $(N = 5, N_\tau = 5)$ in the case of $L = 3$. Correctly predicted scores $s(\widehat{X}, \widehat{Y})$ are indicated in bold.

| label $s(X, Y)$ | X | Y |
|-----------------|--|---|
| 0 |  |  |
| | PMA: 1.000, CATSET: 0.918, CSS: 0.501, PIFR sparse: 0.887, PIFR collaborative: 0.017, bi-PMA: 0.013, pivot-cross: 0.046 | |
| 1 |  |  |
| | PMA: 0.995, CATSET: 0.592, CSS: 0.501, PIFR sparse: 0.670, PIFR collaborative: 0.910, bi-PMA: 0.987, pivot-cross: 0.998 | |
| 0 |  |  |
| | PMA: 0.396, CATSET: 0.502, CSS: 0.501, PIFR sparse: 0.572, PIFR collaborative: 0.360, bi-PMA: 0.003, pivot-cross: 0.009 | |
| 1 |  |  |
| | PMA: 0.995, CATSET: 0.987, CSS: 0.501, PIFR sparse: 0.5671, PIFR collaborative: 0.953, bi-PMA: 0.962, pivot-cross: 1.000 | |

要素ベクトルの変換過程

28

- positiveペア間とnegativeペア間の類似度のヒストグラムの変化



- 集合代表ベクトルsが識別的に変換されていくのがわかる

内容：

30

- 集合データとは
- 様々な順不変関数
- 順不変関数のまとめ
- 集合マッチングへの応用
- 集合検索への発展とまとめ

集合マッチングの課題と集合検索

31

- ポジティブ集合 Y^* を含むギャラリーを**予め用意するのは困難**
 - 運用時には、**未知のクエリ集合 X** が入力される
 - コーデ推薦の場合、クエリはユーザが購入済み（使用中）のアイテムで構成
 - **集合を網羅的に作ると組み合わせ爆発**



家具



ファッション

➡ **集合検索**：予め集合の**ギャラリーは用意せず**、要素ベクトルのデータベース D から**クエリ X に調和するアイテム y^*** を検索し集合 \hat{Y}^* を作成

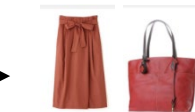
クエリ集合 X ：



カテゴリ集合 Z ：{スカート, バッグ}



Nakamura, et al., arXiv2023



予測集合 \hat{Y}^*

まとめ

32

- 集合 x を扱う関数 $f(x)$ は、**順不変性**が必要
- 順不変性を満たす様々な関数：
 - Deep-sets, PointNet, JanossyPooling, Pooling MAなど
 - Attentionは、全体では**順同変性**、要素では**順不変性**を同時に持つ
- Attentionベースの集合マッチング：
 - Backbone : SABとMABを繰り返して要素ベクトルを変換
 - 凸包上での変換のため、要素数が少ない集合マッチングでは限定的
 - ➡ **Pivot-cross**: クエリをキーとバリューに含めることにより、広い凸包上で変換
 - Head : 順不変性を満たすように、学習ベースのマッチング関数 $f(x, y)$
 - 勾配消失が発生し、深い階層のBackboneの学習が困難
 - ➡ **集合代表ベクトルを導入し、単純な内積により $f(x, y)$ を計算**
- 集合検索への発展：
 - positive集合 Y^* を含むギャリを用意するのは困難
 - ➡ 要素ベクトルのデータベースから、**クエリ集合に合う要素を順次検索し集合 Y^* を動的に作成**