

# This Looks Like That: Deep Learning for Interpretable Image Recognition

Chaofan, C. Oscar, L. Daniel, T. Alina, B. Cynthia, R. Jonathan, K.  
In Advances in Neural Information Processing Systems 32 (NeurIPS 2019)

# At first

---

What is this animal ?



# Introduction

- 人間が識別するプロセスと近い説明を与えるネットワーク **Prototypical Part Network (ProtoPNet)** を提案
- ProtoPNetは明確な推論プロセスを持つ点で解釈可能である (**interpretable**)
- 既存のDeep learning model (non-interpretable) と同等な学習精度を達成していることを実験的に確認
- 他のinterpretableなモデルには無い解釈性を提供



finding prototypical parts



combines evidence from  
the prototypes



make a classification

# Introduction

## What is prototype ?

一般的な意味として, 各prototypeは訓練データの一部を代表する抽象的な表現つまり, 全てのprototypeは全訓練データを表現する抽象的な表現となっている

本論文では



予め各クラスに対して複数のprototypeが割り当てられており,  
クラス $k \in \{1, \dots, K\}$ に割り当たる各prototypeは, クラス $k$ であると表現される  
局所的な表現となっている  
分類時には, 学習されたprototypeと入力画像の類似度を比較して推論を行う

input



four prototypes of sparrow class



類似度を計算

分類



# Introduction

## ProtoPNet' Classification Process

finding prototypical parts



combines evidence from  
the prototypes

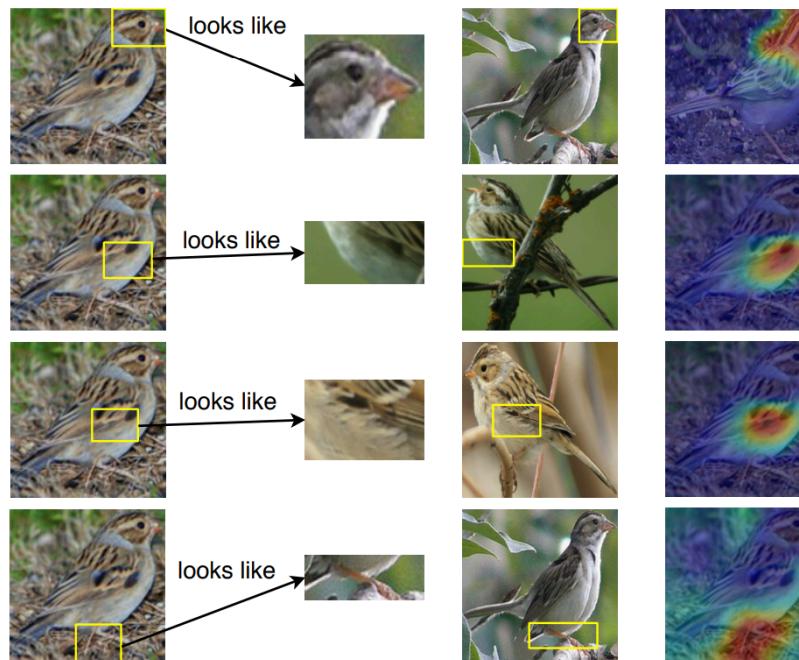


make a classification

test image



prototypes



*Leftmost:* a test image of a clay-colored sparrow  
*Second column:* same test image, each with a  
bounding box generated by our model  
-- the content within the bounding box  
is considered by our model to look similar  
to the prototypical part (same row, third  
column) learned by our algorithm

*Third column:* prototypical parts learned by our  
algorithm

*Fourth column:* source images of the prototypical  
parts in the third column

*Rightmost column:* activation maps indicating how  
similar each prototypical part resembles  
part of the test bird

Figure 1: Image of a clay colored sparrow and how parts of it look like some learned prototypical parts of a clay colored sparrow used to classify the bird's species.

# Introduction

---

## Related work

### post-hoc interpretability analysis (ex. GradCAM, SmoothGrad, ... )

- ✓ 訓練されたモデルに対して、説明性を与える手法
- ✓ ネットワークが実際にどのように決定をするかの推論プロセスを説明するものではない
- ➡ 本論文では、推論プロセスを含めた説明性を与える

### attention-based interpretability (Ex. Part R-CNN, PS-CNN, 2-level attn, Neural const, RA-CNN, ...)

- ✓ 決定するときに注目している箇所を強調する手法
- ✓ 入力のどの部分を注目しているのかを示すのみ
- ➡ 注目しているのかを示すことに加えて、それらの部分に類似した典型的なケースを示している

### other prototype classification techniques

**prototype** (≒訓練データの一部を代表する表現)を用いた学習

- ✓ 本論文は[24]と最も近い
  - ✓ prototypeを可視化するためにdecoderが必要
  - ✓ 自然画像の場合、可視化の際に現実的なprototypeの作成に失敗
- ➡ ✓ decoder不要
  - ✓ 全てのprototypeはある訓練データの潜在表現のため、忠実に可視化可能

# Case study 1: bird species identification

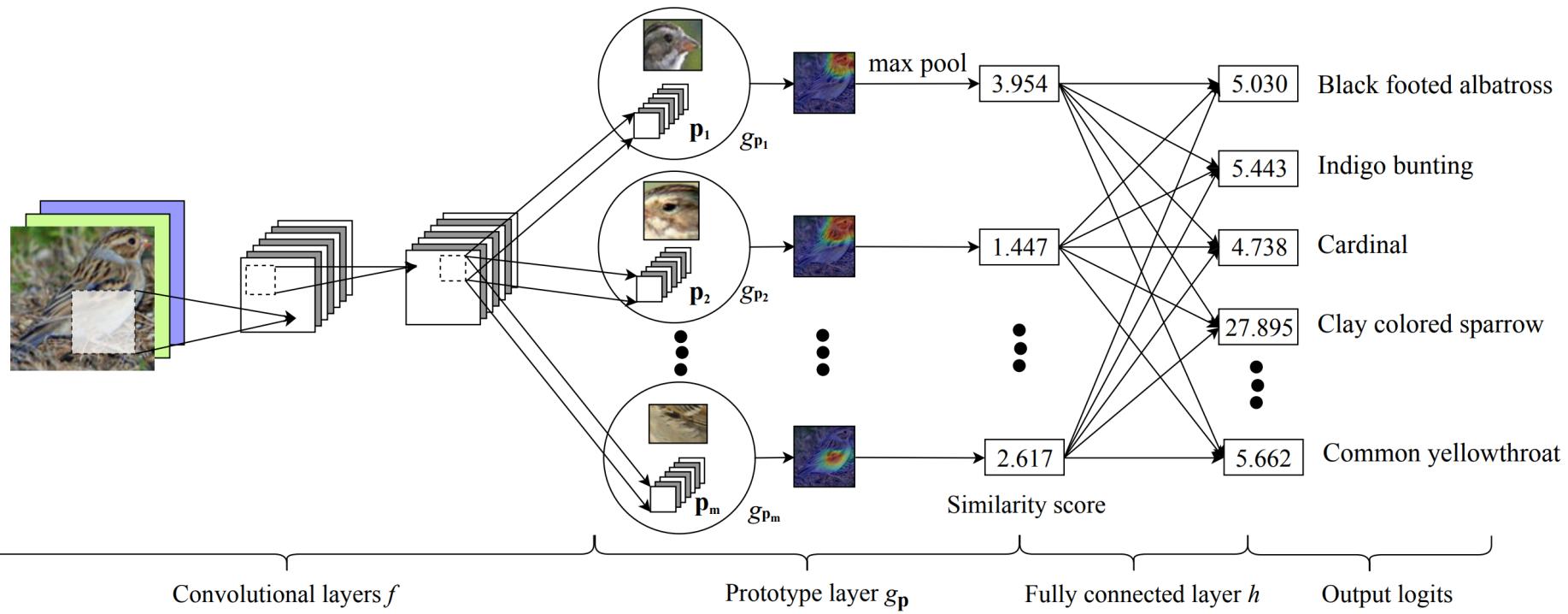
Convolution layers :  $f$

parameters :  $w_{\text{conv}}$

activation function (except last layer) : ReLU  
activation function (last layer) : sigmoid

Prototype layers :  $g_p$

Fully connected layers :  $h$   
parameters :  $w_h$   
no bias



# Case study 1: bird species identification

入力画像を潜在空間に射影

Convolution layers :  $f$

parameters :  $w_{\text{conv}}$

activation function (except last layer) : ReLU

activation function (last layer) : sigmoid

Prototype layers :  $g_p$

Fully connected layers :  $h$

parameters :  $w_h$

no bias



input :  $x$

$224 \times 224 \times 3$



output :  $z = f(x)$   
 $7 \times 7 \times D$

※  $D$  is chosen from three possible values: 128, 256, 512, using cross validation)

# Case study 1: bird species identification

潜在空間内の入力画像とPrototypeの距離を計算

Convolution layers :  $f$

parameters :  $w_{\text{conv}}$

activation function (except last layer) : ReLU

activation function (last layer) : sigmoid

Prototype layers :  $g_p$

Fully connected layers :  $h$

parameters :  $w_h$

no bias



input :  $z = f(x)$   
 $H \times W \times D (= 7 \times 7 \times D)$



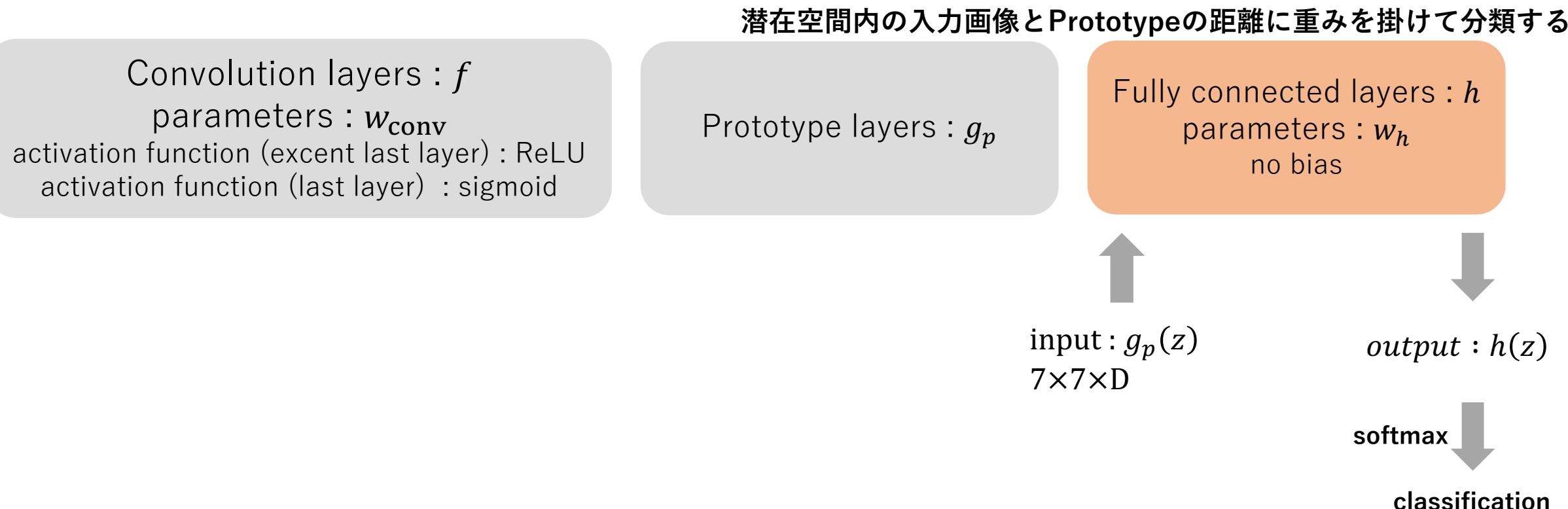
output :  $g_p(z)$   
 $\{g_{p_i}(z)\}_{i=1}^m \in g_p(z)$

- ✓  $m$  個 prototype :  $P = \{p_j\}_{j=1}^m, H_1 \times W_1 \times D (= 1 \times 1 \times D)$
- ✓  $H_1 \leq H, W_1 \leq W \rightarrow$  それぞれの prototype  $p_j$  は, 元の空間で, 画像のある **patch** ( $\Leftarrow$ 一部分) を表す
- ✓  $f(x)$  と  $j$  番目の prototype  $p_j$  の  $L_2$  距離に基づくスコア  $g_{p_i}(z)$  を計算 ( $z = f(x)$  とする)

$$g_{p_j}(z) = \max_{\tilde{z} \in patches(z)} \log \frac{\left( \|\tilde{z} - p_j\|_2^2 + 1 \right)}{\left( \|\tilde{z} - p_j\|_2 + \epsilon \right)}$$

- $g_{p_i}(z)$  が大きいほど,  $j$  番目の prototype と似ている (近い) patch が入力画像内にあるということ  
 $g_{p_i}(z)$  に対して, max pooling を行うことで類似度 (**Similarity score**) を算出できる
- ✓ それぞれのクラス  $k \in \{1, \dots, K\}$  に対して  $m_k$  個の prototype が割り当てられている

# Case study 1: bird species identification



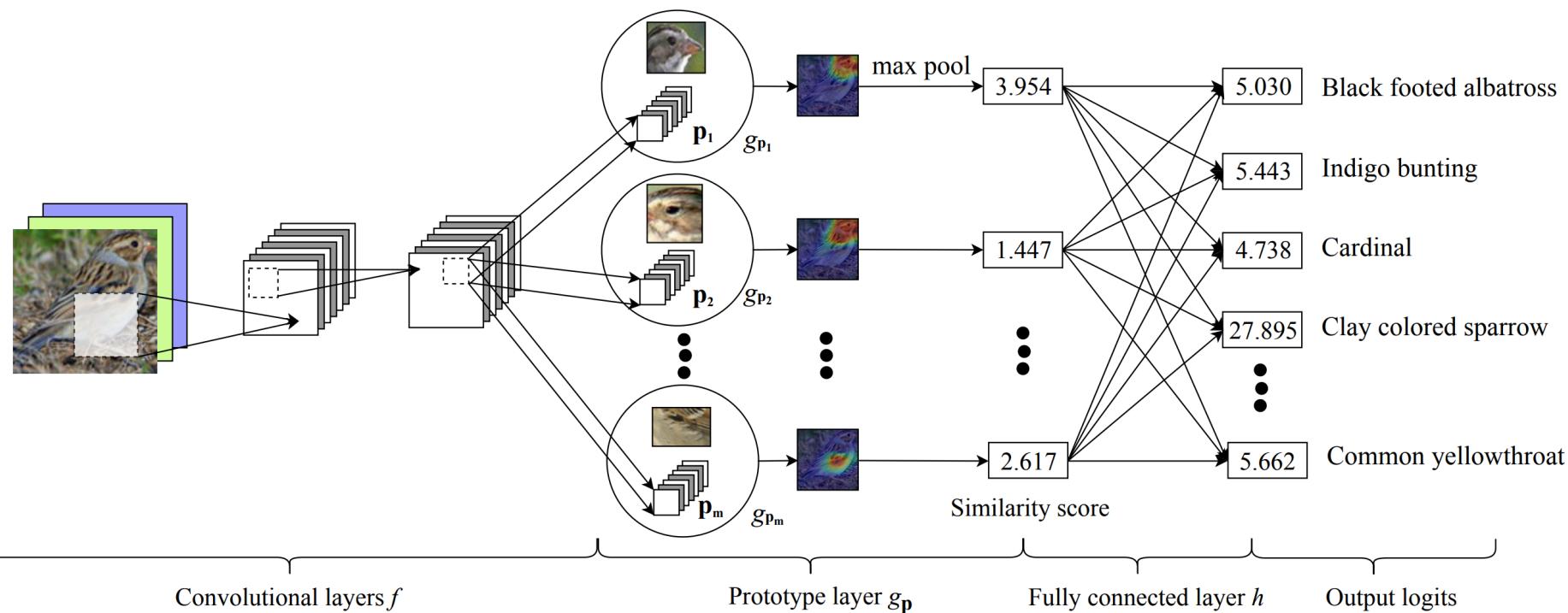
- ✓  $h(z) = w_h g_p(z)$  を求める
- ✓ あるクラスのprototypeとの類似度に関する重みが学習される
- ➡ 分類におけるそれぞれのprototypeの重要度を求めることができる

※ ProtoPNetの推論メカニズムはいくつかの合理的な仮定のもとで、より一般的な確率的推論 (probabilistic inference) とみなすことができる。  
これについては補足資料のS2を参照

# Case study 1: bird species identification

## Training algorithm

- (1) stochastic gradient descent (SGD) of layers before the last layer
- (2) projection of prototypes
- (3) convex optimization of last layer



# Case study 1: bird species identification

## Training algorithm

### (1) stochastic gradient descent (SGD) of layers before the last layer

- ある訓練データに対して、最も重要なpatchと自身のクラスのprototypeが潜在空間内の距離が近くなり他のクラスのprototypeが十分に分離されるような $f$ を学習する

#### object function

$$\min_{\mathbf{P}, w_{\text{conv}}} \frac{1}{n} \sum_{i=1}^n \text{CrsEnt}(h \circ g_{\mathbf{p}} \circ f(\mathbf{x}_i), \mathbf{y}_i) + \lambda_1 \text{Clst} + \lambda_2 \text{Sep}, \quad \text{where Clst and Sep are defined by}$$

$$\text{Clst} = \frac{1}{n} \sum_{i=1}^n \min_{j: \mathbf{p}_j \in \mathbf{P}_{y_i}} \min_{\mathbf{z} \in \text{patches}(f(\mathbf{x}_i))} \|\mathbf{z} - \mathbf{p}_j\|_2^2; \text{Sep} = -\frac{1}{n} \sum_{i=1}^n \min_{j: \mathbf{p}_j \notin \mathbf{P}_{y_i}} \min_{\mathbf{z} \in \text{patches}(f(\mathbf{x}_i))} \|\mathbf{z} - \mathbf{p}_j\|_2^2.$$

cross entropy loss(分類誤差)を小さくすることを要請

各訓練データのpatchは少なくとも1つの自身のクラスのprototypeに近いことを要請

各訓練データのpatchは他のクラスのprototypeから離れることを要請

# Case study 1: bird species identification

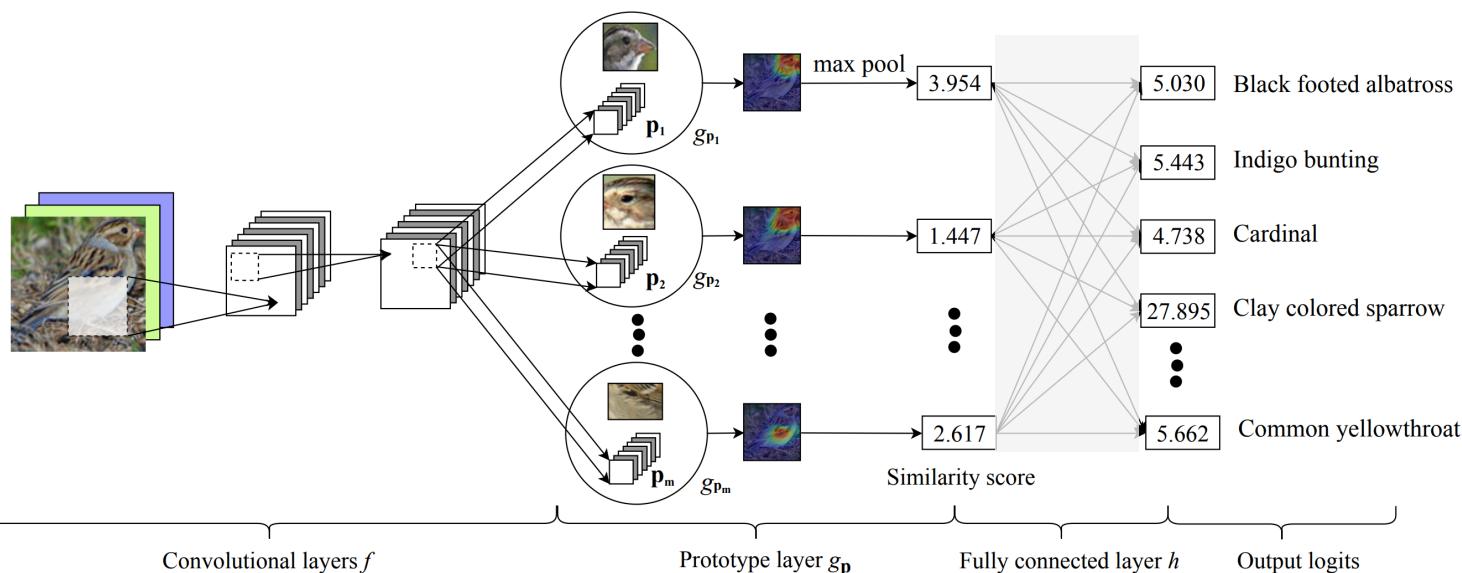
## Training algorithm

### (1) stochastic gradient descent (SGD) of layers before the last layer

- ▶ 全結合層の重みを以下のようにして、学習する

$$w_h^{(k,j)} = 1 \text{ for all } j \text{ with } p_j \in P_k, \quad w_h^{(k,j)} = -0.5 \text{ for all } j \text{ with } p_j \notin P_k$$

- ▶  $y \neq k$ である  $x$ に対して、 $f(x)$ と  $p_j \in P_k$  の距離が近い場合に分類精度が悪化  
互いに異なるクラスの prototype 同士が離れるように学習される



notation	description
$(x, y)$	訓練データ, 正解データ
$P_k \subseteq P$	クラス $k$ の prototype 集合
$w_h^{(k,j)}$	$j$ 番目の prototype unit と クラス $k$ の ロジット の 重み

# Case study 1: bird species identification

## Training algorithm

### (2) projection of prototypes

- prototype  $p_j$ を可視化するために  $p_j$ と同クラスで潜在空間内で最も近い訓練データのpatchに置き換える

$$\mathbf{p}_j \leftarrow \arg \min_{\mathbf{z} \in \mathcal{Z}_j} \|\mathbf{z} - \mathbf{p}_j\|_2, \text{ where } \mathcal{Z}_j = \{\tilde{\mathbf{z}} : \tilde{\mathbf{z}} \in \text{patches}(f(\mathbf{x}_i)) \ \forall i \text{ s.t. } y_i = k\}.$$

- ✓ prototypeの射影が分類精度にどのような影響を及ぼすか理論的に述べる
- ➡ prototypeの射影がprototypeをあまり動かさないならば, 予測結果は変化しない (**Theorem 2.1.**)

# Case study 1: bird species identification

## Theorem 2.1.

Let

notation	description
$b_l^k$	射影前のクラス $k$ の $l$ 番目の prototype
$a_l^k$	射影後のクラス $k$ の $l$ 番目の prototype
$x$	入力画像
$c$	$x$ の正解ラベル
$z_l^k$	$b_l^k$ に最も近い $f(x)$ の patch $(\operatorname{argmin}_{\tilde{z} \in \text{patches}(f(x))} \ \tilde{z} - b_l^k\ _2)$

Suppose

- i. 以下を満たす  $0 < \delta < 1$  が存在する
  - a. 全ての異なるクラス  $k \neq c$  の prototypeにおいて,  
$$\|a_l^k - b_l^k\|_2 \leq \theta \|a_l^k - b_l^c\|_2 - \sqrt{\epsilon}, \text{ where } \theta = \min\left(\sqrt{1 + \delta} - 1, 1 - \frac{1}{\sqrt{2 - \delta}}\right)$$
  - b. 同一クラス  $c$  の prototypeにおいて,  
$$\|a_l^c - b_l^c\|_2 \leq (\sqrt{1 + \delta} - 1) \|z_l^c - b_l^c\|_2 \text{ and } \|z_l^c - b_l^c\|_2 \leq \sqrt{1 - \delta}$$
- ii. それぞれのクラスの prototype の数は等しい ( $m'$ )
- iii. 最終層の重みは以下となる  
$$w_h^{(k,j)} = 1 \text{ with for all } j \text{ with } p_j \in P_k, w_h^{(k,j)} = 0 \text{ with for all } j \text{ with } p_j \notin P_k$$

Then

射影後, 正しいクラス  $c$  のロジットは最大で  $\Delta_{\max} = m' \log((1 + \delta)(2 - \delta))$  減少し,

全ての異なるクラス  $k \neq c$  のロジットは最大で  $\Delta_{\max}$  増加する

→ 上位 2 クラスのロジットが少なくとも  $2\Delta_{\max}$  離れているならば, prototype の射影は  $x$  の予測を変更しない

# Case study 1: bird species identification

---

## Training algorithm

### (3) convex optimization of last layer

- $p_j \notin P_k$  の  $k$  と  $j$  の接続  $w_h^{(k,j)}$  を  $w_h^{(k,j)} \approx 0$  にしたい
- ➡ 否定的な推論 (ex. この鳥はクラス  $k$  の prototype に近い patch を持っていないから クラス  $k$  である)を行いたくない

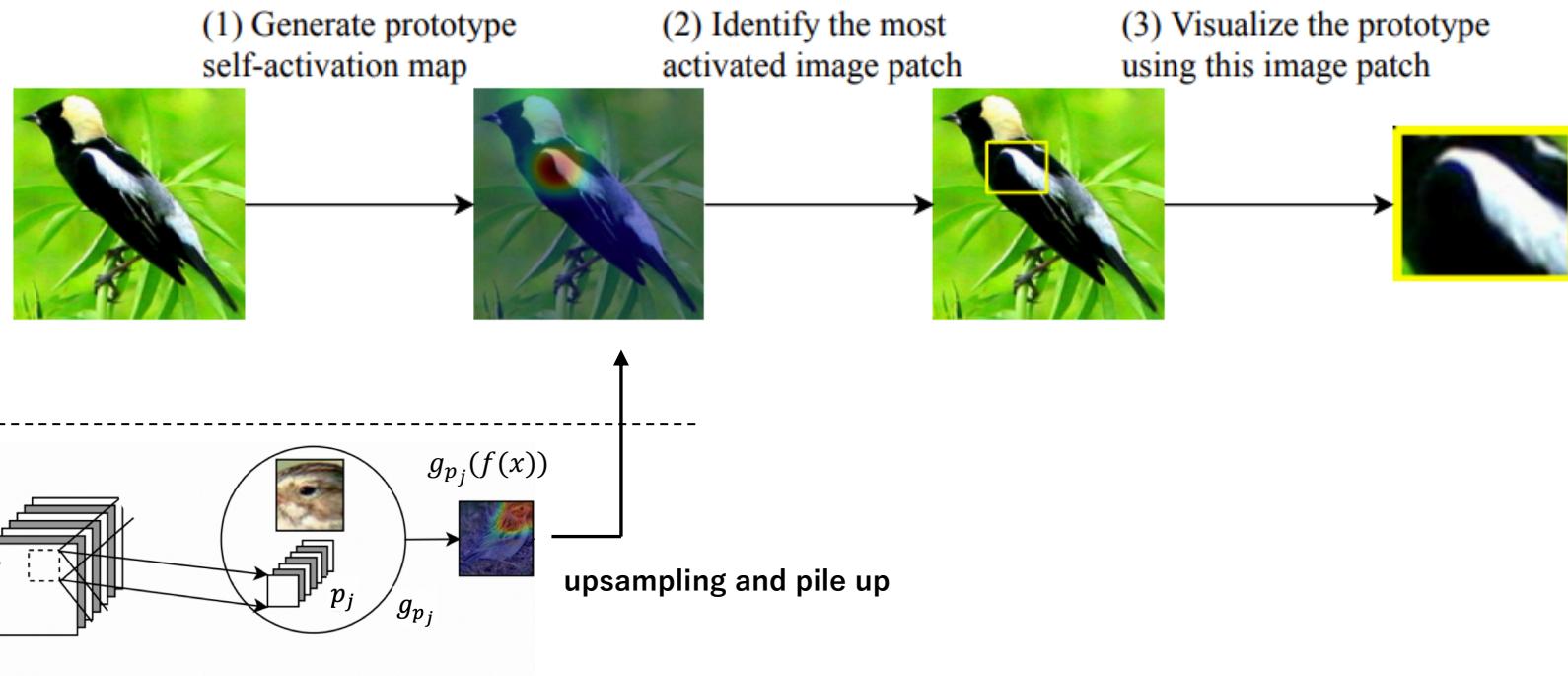
## object function

$$\min_{w_h} \frac{1}{n} \sum_{i=1}^n \text{CrsEnt}(h \circ g_{\mathbf{P}} \circ f(\mathbf{x}_i), \mathbf{y}_i) + \lambda \sum_{k=1}^K \sum_{j: \mathbf{p}_j \notin \mathbf{P}_k} |w_h^{(k,j)}|.$$

# Case study 1: bird species identification

## Prototype visualization

- デコーダーを用いずに、潜在空間上のprototype  $p_j$ を元の画像空間に戻して可視化する
  - ✓ prototype  $p_j$ は $f(x)$ に射影されているため、 $p_j$ による $x$ のactivation mapを利用して $p_j$ を可視化する
- (1) activation mapは $7 \times 7$ なので、 $p_j$ による $x$ のactivation mapを元の次元 $224 \times 224$ にupsamplingを行う
  - (2) activation mapの合計ピクセル値に対して95%以上の割合を囲める最小な四角形を求める
  - (3) 得られた四角形を切り取る



※ 論文中の別々の画像を使用しているため、画像の対応が取れていない

# Case study 1: bird species identification

## Reasoning process of our network

- テストデータ $x$ に対するシマセゲラ (red-bellied woodpecker)の推論プロセス
- ✓ prototype  $p_i$ と $f(x)$ のpatchを比較し, similarity scoreを求め,  
シマセゲラに対応する最終層の重みをかけることでprototype  $p_i$ 求めることができる (**points contributed**)  
points contributedの和を取ることで, テストデータ $x$ がシマセゲラに属するスコアを求めることができる

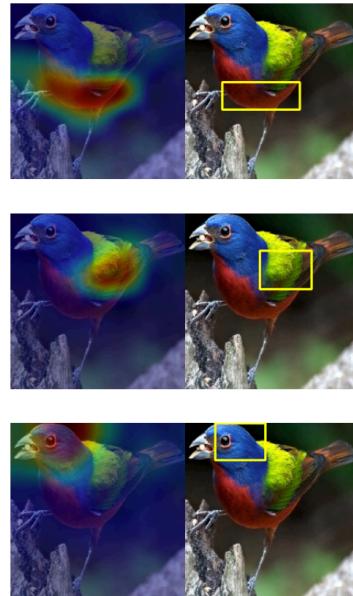


# Case study 1: bird species identification

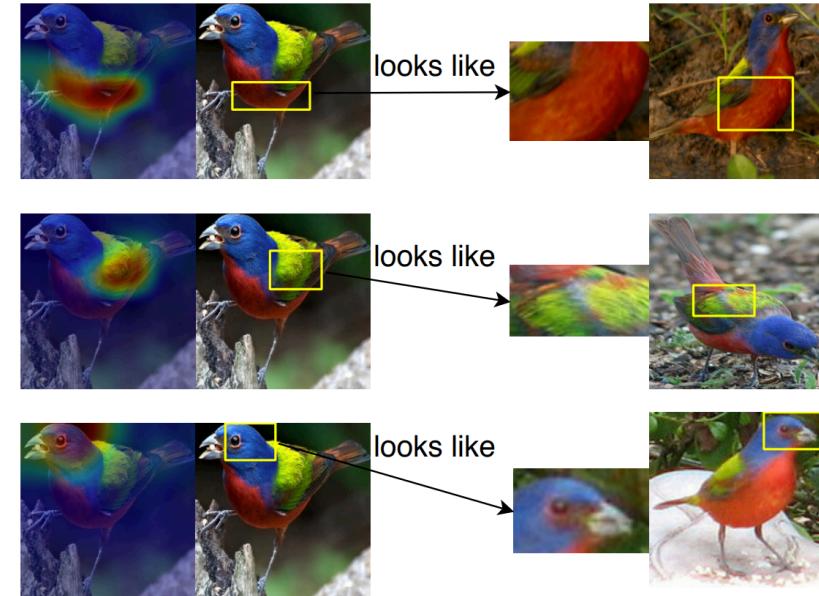
Comparison with baseline models and attention-based interpretable deep models



(a) Object attention  
(class activation map)



(b) Part attention  
(attention-based models)



(c) Part attention + comparison with learned  
prototypical parts (our model)

Figure 4: Visual comparison of different types of model interpretability: (a) object-level attention map (e.g., class activation map [53]); (b) part attention (provided by attention-based interpretable models); and (c) part attention with similar prototypical parts (provided by our model).

# Case study 1: bird species identification

## Comparison with baseline models and attention-based interpretable deep models

- ProtoPNetとBaseline (without the prototype layer)との精度比較

Base	ProtoPNet	Baseline	Base	ProtoPNet	Baseline
VGG16	$76.1 \pm 0.2$	$74.6 \pm 0.2$	VGG19	$78.0 \pm 0.2$	$75.1 \pm 0.4$
Res34	$79.2 \pm 0.1$	$82.3 \pm 0.3$	Res152	$78.0 \pm 0.3$	$81.5 \pm 0.4$
Dense121	$80.2 \pm 0.2$	$80.5 \pm 0.1$	Dense161	$80.1 \pm 0.3$	$82.2 \pm 0.2$

- ProtoPNetと既存手法との精度比較

full : model was trained on full images

bb : model was trained on images cropped usinng bounding boxes (or the model used bounding boxes in other ways)

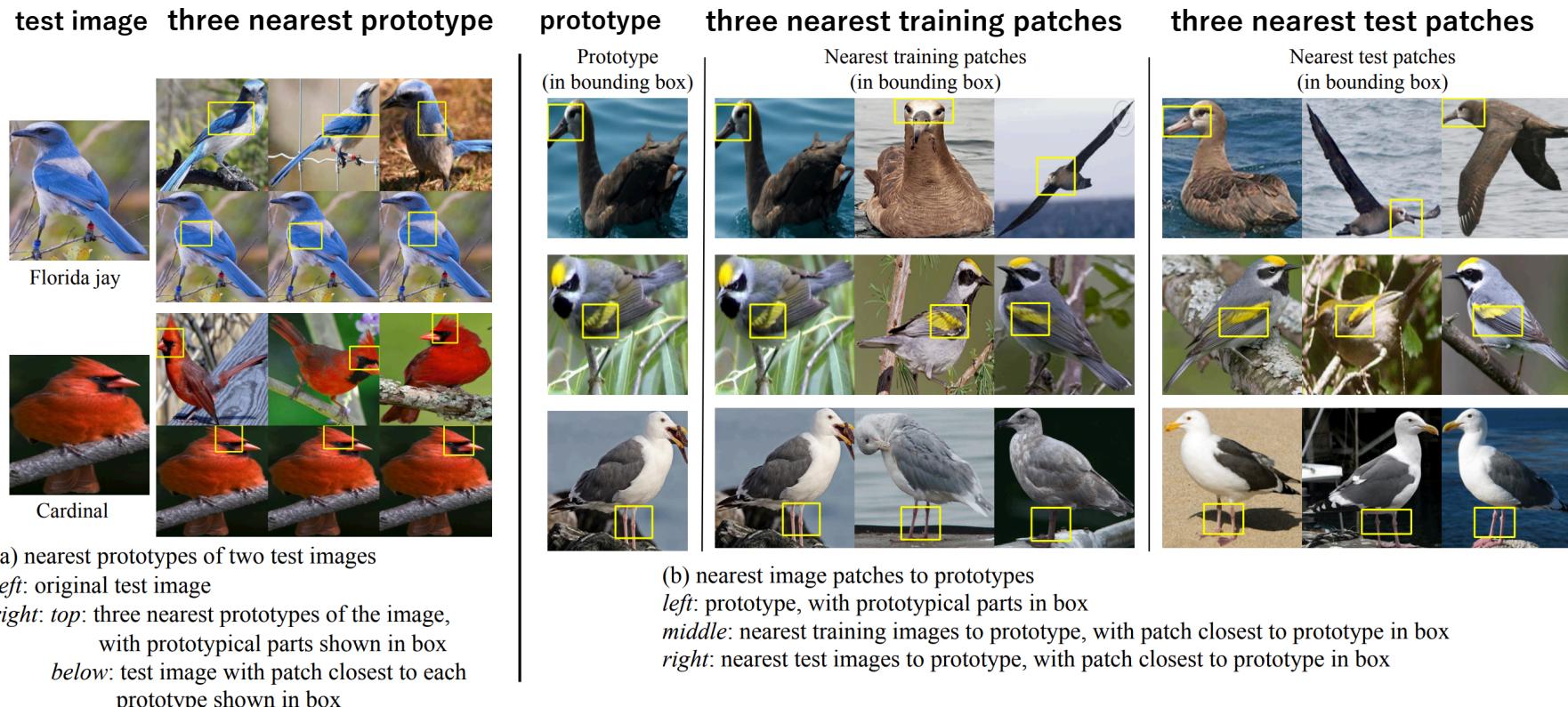
anno : model was trained with keypoint annotations of bird parts

Interpretability	Model: accuracy
None	<b>B-CNN</b> [26]: 85.1 (bb), 84.1 (full)
Object-level attn.	<b>CAM</b> [53]: 70.5 (bb), 63.0 (full)
Part-level attention	<b>Part R-CNN</b> [50]: 76.4 (bb+anno.); <b>PS-CNN</b> [16]: 76.2 (bb+anno.); <b>PN-CNN</b> [3]: 85.4 (bb+anno.); <b>DeepLAC</b> [25]: 80.3 (anno.); <b>SPDA-CNN</b> [49]: 85.1 (bb+anno.); <b>PA-CNN</b> [20]: 82.8 (bb); <b>MG-CNN</b> [45]: 83.0 (bb), 81.7 (full); <b>ST-CNN</b> [17]: 84.1 (full); <b>2-level attn.</b> [46]: 77.9 (full); <b>FCAN</b> [27]: 82.0 (full); <b>Neural const.</b> [36]: 81.0 (full); <b>MA-CNN</b> [52]: 86.5 (full); <b>RA-CNN</b> [8]: 85.3 (full)
Part-level attn. + prototypical cases	<b>ProtoPNet</b> (ours): 80.8 (full, VGG19+Dense121+Dense161-based) 84.8 (bb, VGG19+ResNet34+DenseNet121-based)

# Case study 1: bird species identification

## Analysis of latent space and prototype pruning

- ある検証データのpatchに対して近傍にあるprototypeの上位3件
- あるprototypeに対して近傍にあるpatchの上位3件



- ✓ 検証データの近傍にあるprototypeは同一のクラスからなる
- ✓ あるprototypeのpatchは同質の意味概念を持つ

# Conclusion

---

- ✓ 人間が画像を分類するプロセスに従った解釈性を与えるネットワーク  
ProtoPNetを提案
- ✓ ProtoPNetを 2 つのデータセットに適用して性能の検証を行った  
※Case study 2はsupplementで詳細に述べられていたため、本資料では省略