This Looks Like That: Deep Learning for Interpretable Image Recognition

02/07/2020

Presenter: Zijie Pan

https://qdata.github.io/deep2Read/

Motivation

- How human classify images? : We would focus on parts of the image and compare them with prototypical parts of images from a given class.

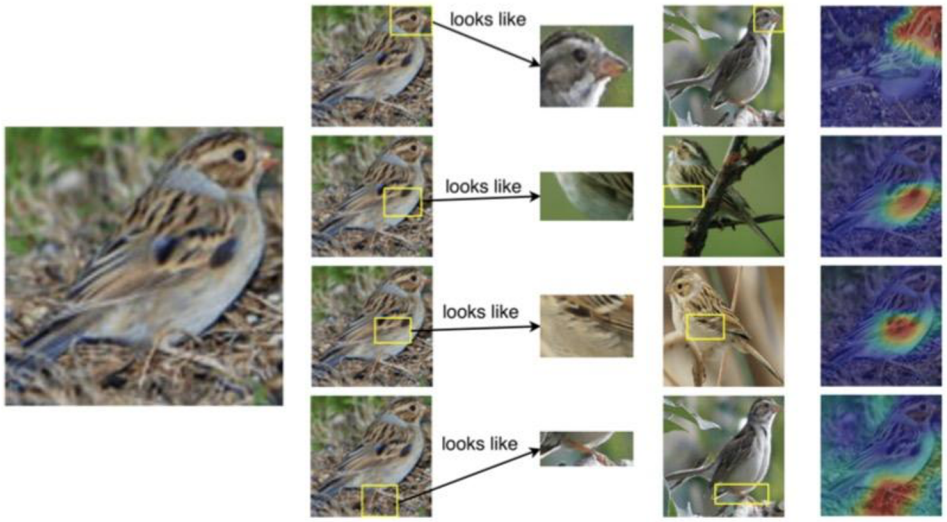
- Apply this decision making to neural network classification so that the model is interpretable

Claim / Target Task

- Propose the architecture of *prototypical part network* (ProtoPNet)
- Propose a form of interpretability (This Looks Like That)

- The model is able to identify several parts of the image where it thinks that this identified part of the image looks like that prototypical part of some training image, and makes its prediction based on a weighted combination of the similarity scores

An Intuitive Figure Showing WHY Claim



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.

Model Architecture

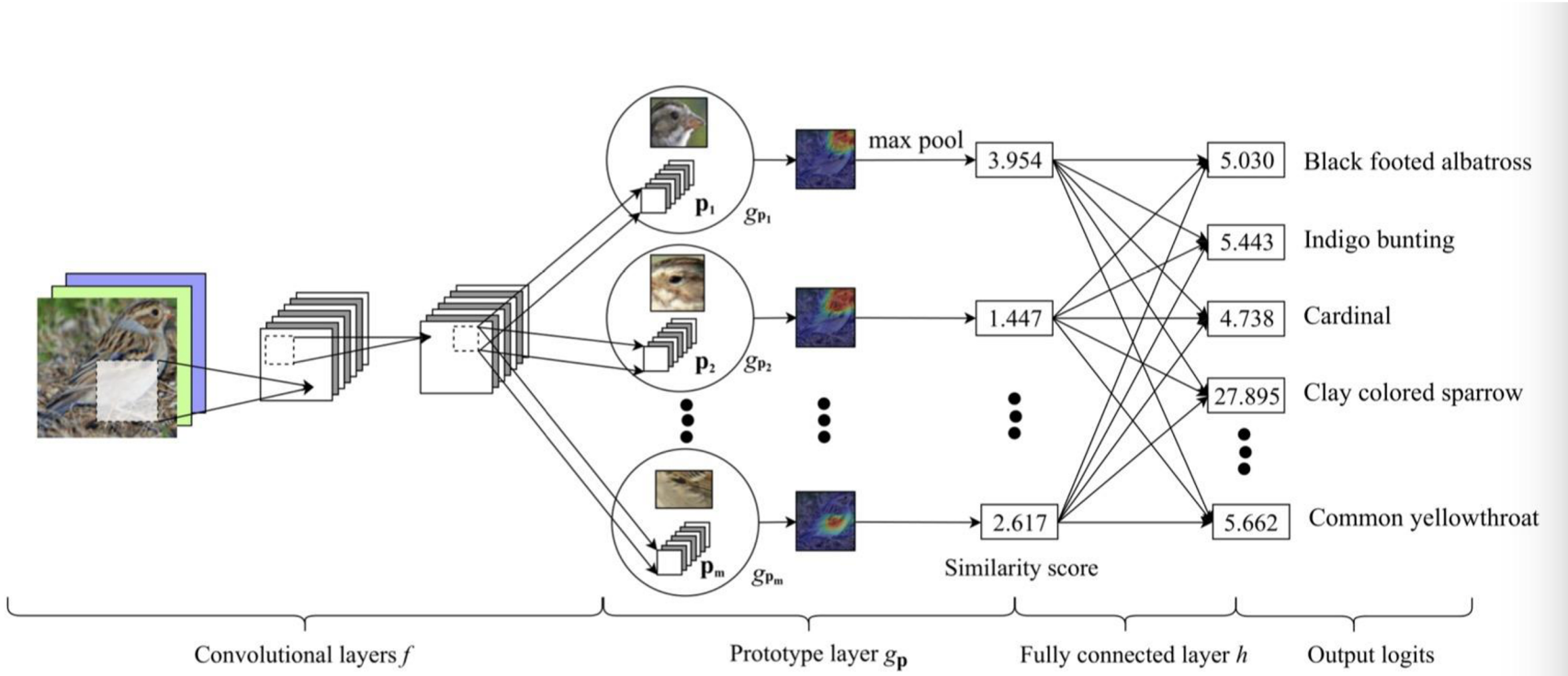


Figure 2: ProtoPNet architecture.

Model Architecture Cont.

- Initially we have the first 13 layers of the VGG-16 (or other existing)network ,followed by 2 , 1x1 convs
- The network learns m prototypes (pj) , with a predefined number of prototypes for each class .
- Each prototype is applied to all the patches of the conv output , and using the L2 distance , a similarity score is produced for a single prototype for all the patches , this can also be used to make a heatmap of similarity .
- Then global pooling is applied to convert this into a single score gpj.Which represents the strongest similar finding of that prototype in the image.
- Then the m global similarity scores are feeded in the FC layer to have weight combination to perform classification.

Classification Process

Training Algorithm

- **SGD of the layers before the FC layer h** :This aims to learn a meaningful latent space where the most important patches for classifying images are clustered around prototypes associated with their own classes (Clustering Cost), and those important patches from different classes will be separated into distinct clusters(Separation Cost) . The loss function is sum of the cross entropy loss , the clustering cost and the separation cost .

- **Projection of the prototypes onto the closest latent representations of training image patches from the same class as that of the prototype**. :We push each prototype pj onto the closest latent representation of training image patches from the same class as that of pj . So effectively each prototype now corresponds to some patch of a image in the training set

- **Convex optimization of the last layer h**.- The parameters of the conv , and the prototype layers are fixed . Also the weights of the last layer are encouraged to be sparse by adding L1 regularisation.

Training Algorithm

- SGD Training Before FC layer

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

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

Experimental Analysis

- Prototype Projection

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

- FC Layer Parameter Training

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

Experiment Results:

Test Dataset: cropped bird images of CUB-200-2011

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

Interpretability	Model: accuracy
None	**B-CNN**[25]: 85.1 (bb), 84.1 (full)
Object-level attn.	**CAM**[56]: 70.5 (bb), 63.0 (full)
Part-level attention	**Part R-CNN**[53]: 76.4 (bb+anno.); **PS-CNN** [15]: 76.2 (bb+anno.); **PN-CNN** [3]: 85.4 (bb+anno.); **DeepLAC**[24]: 80.3 (anno.); **SPDA-CNN**[52]: 85.1 (bb+anno.); **PA-CNN**[19]: 82.8 (bb); **MG-CNN**[46]: 83.0 (bb), 81.7 (full); **ST-CNN**[16]: 84.1 (full); **2-level attn.**[49]: 77.9 (full); **FCAN**[26]: 82.0 (full); **Neural const.**[37]: 81.0 (full); **MA-CNN**[55]: 86.5 (full); **RA-CNN**[7]: 85.3 (full)
Part-level attn. + prototypical cases	**ProtoPNet** (ours): 80.8 (full, VGG19+Dense121+Dense161-based) 84.8 (bb, VGG19+ResNet34+DenseNet121-based)

Comments:

Strength: The classification process is very interpretable and can reach comparable accuracy with its standard non-interpretable counterpart as well as other interpretable deep models

Weakness: What if prototypes lack of representational power

What if similar prototypes from other classes can also create high similarity scores, may cause misclassification.