Learning by Comparison: Case-based Reasoning for Interpretable Vision Models

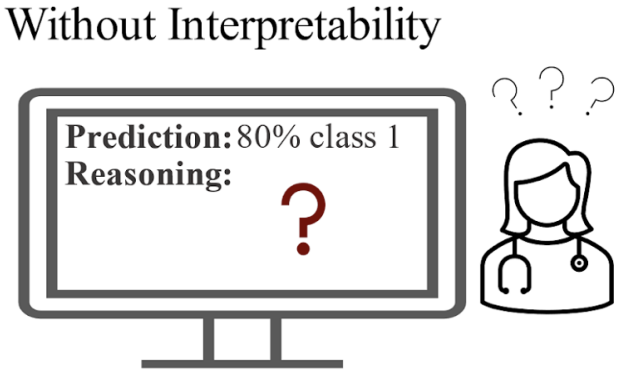
Dr. Chaofan Chen
Assistant Professor
Electrical and Computer Engineering

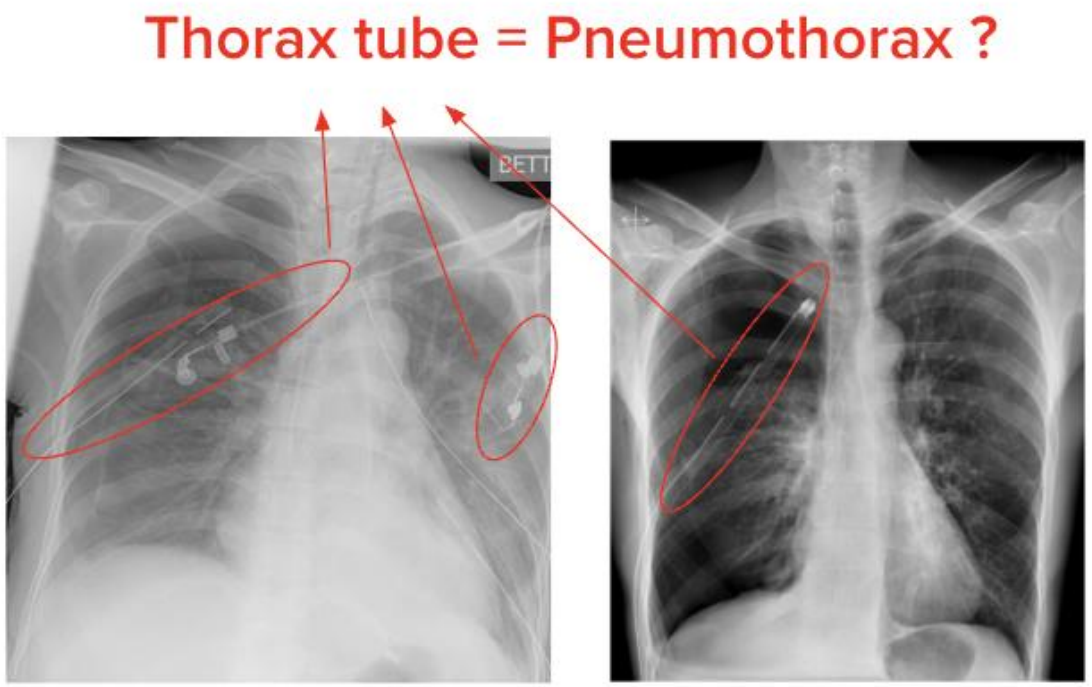
1865 THE UNIVERSITY OF MAINE

Why interpretability?

Black-box model:

- **Opaque reasoning**: blind to model's inner working
- **Cannot explain its predictions**: no way to verify its predictions
- **Could be right for the wrong reason**: no way to assess whether model is using relevant evidence or confounders





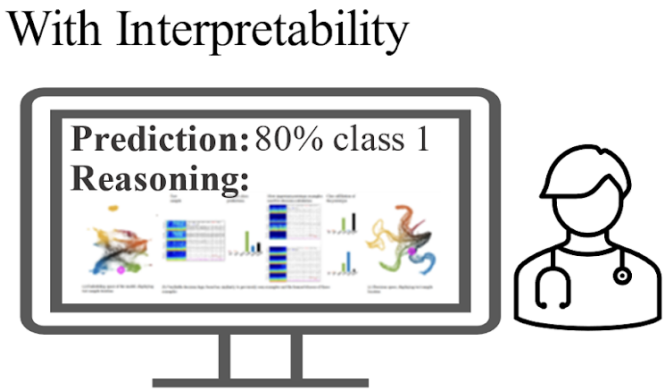
Using the presence of thorax tubes to detect pneumothorax in chest X-rays

Rueckel et al. "Impact of confounding thoracic tubes and pleural dehiscence extent on artificial intelligence pneumothorax detection in chest radiographs." Investigative Radiology 55.12 (2020): 792-798.

Why interpretability?

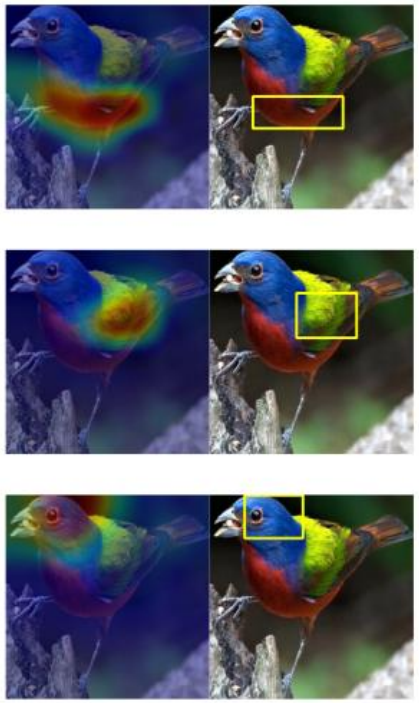
Interpretable model:

- **Transparent reasoning**: humans can understand model's inner working
- **Can explain its predictions**: able to see supportive evidence for predictions, and disregard predictions where model's reasoning is wrong
- **Could be steered toward right reasoning**



Interpretability in computer vision

(a) Object attention
(class activation map)



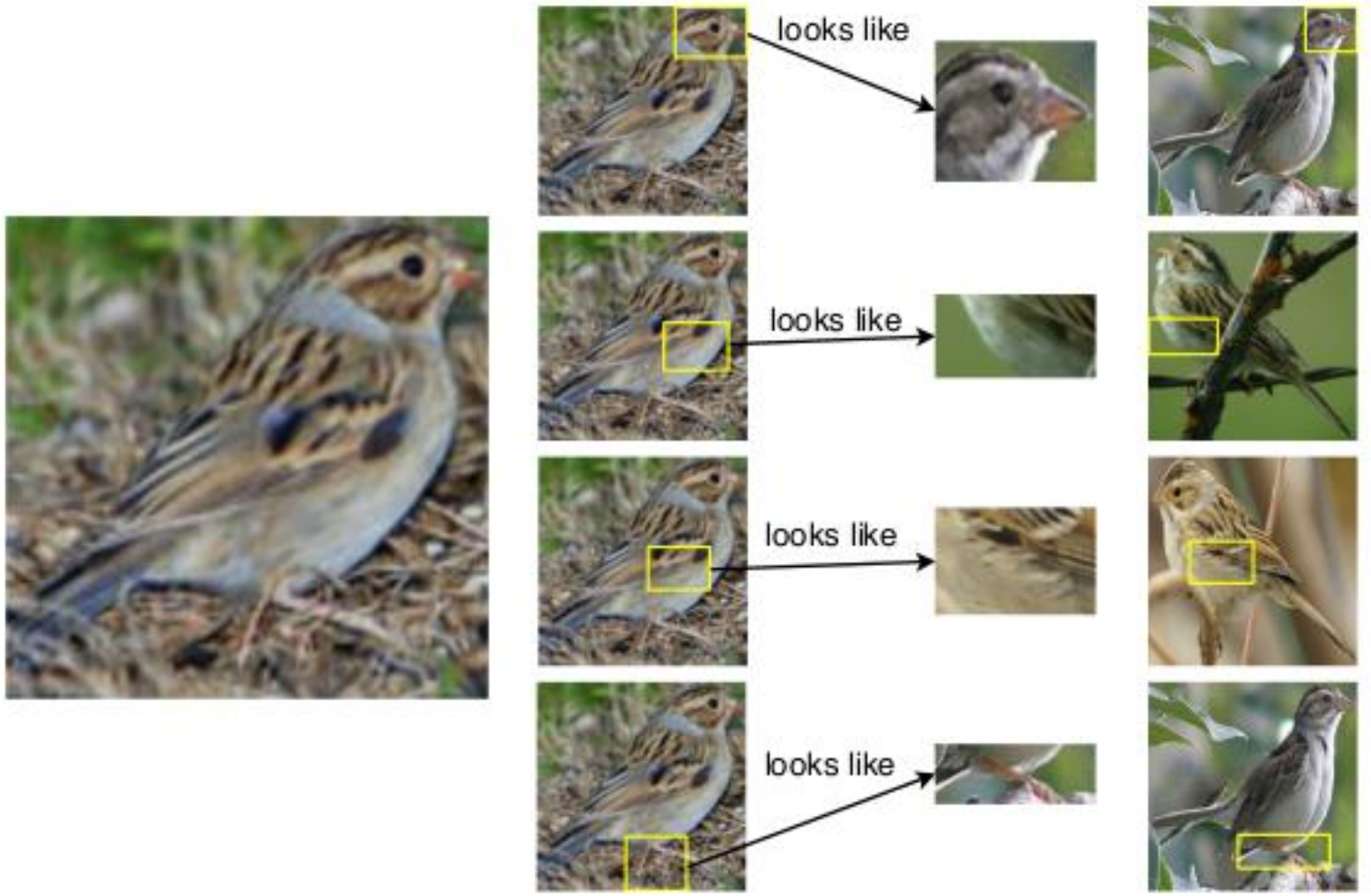
(b) Part attention
(attention-based models)

In the past:

- Synonymous with “pixel attributions”
- What are the important pixels that contribute to a prediction?

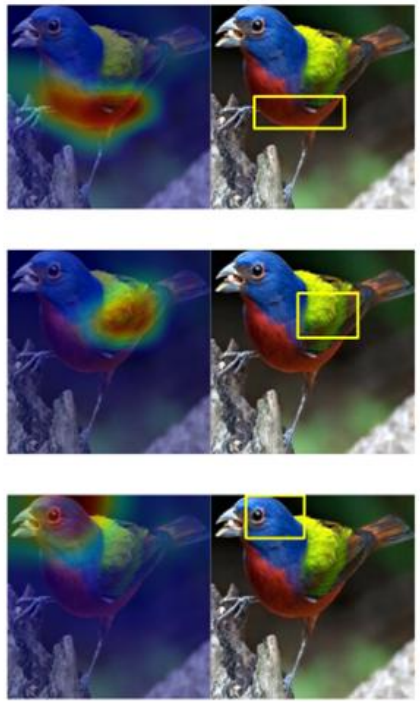
A new form of interpretability: “*this looks like that*”

A new form of interpretability: "*this looks like that*"

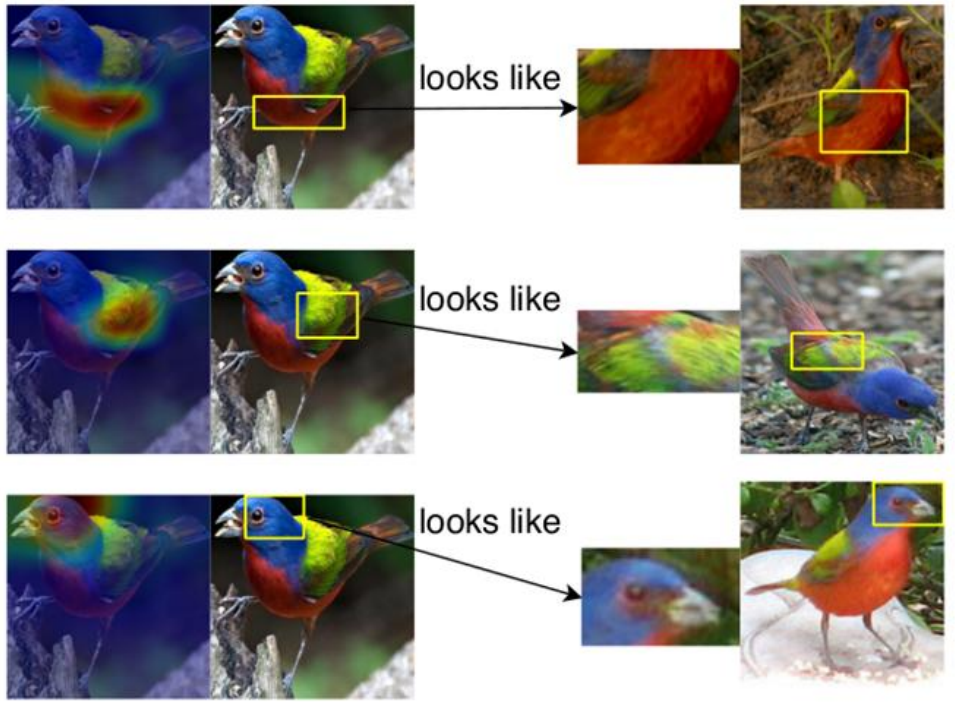


A new form of interpretability: "*this looks like that*"

(a) Object attention
(class activation map)



(b) Part attention
(attention-based models)



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

Prototypical part network (ProtoPNet)

- “This Looks Like That: Deep Learning for Interpretable Image Recognition” (NeurIPS, 2019)

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

Chaofan Chen*
Duke University
cfchen@cs.duke.edu

Oscar Li*
Duke University
oscarli@alumni.duke.edu

Chaofan Tao
Duke University
chaofan.tao@duke.edu

Alina Jade Barnett
Duke University
abarnett@cs.duke.edu

Jonathan Su
MIT Lincoln Laboratory†
su@ll.mit.edu

Cynthia Rudin
Duke University
cynthia@cs.duke.edu

Abstract

When we are faced with challenging image classification tasks, we often explain our reasoning by dissecting the image, and pointing out prototypical aspects of

Prototypical part network (ProtoPNet)

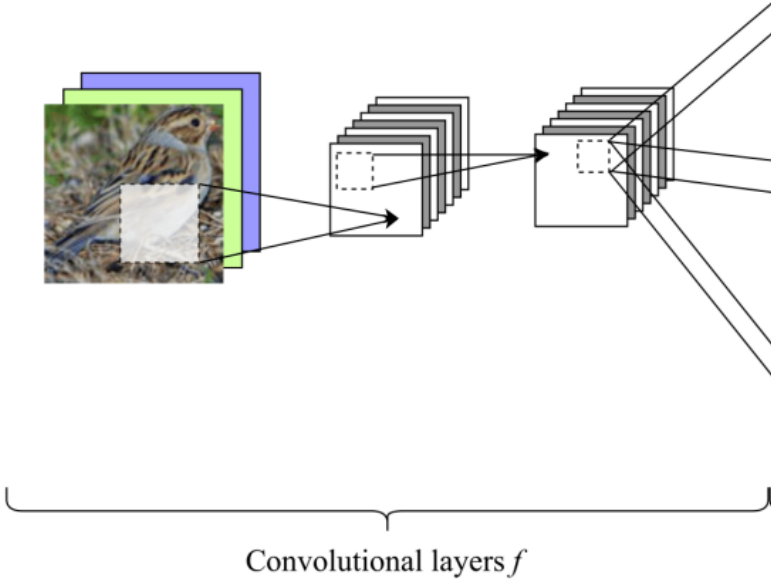


Figure 2: ProtoPNet architecture.

Prototypical part network (ProtoPNet)

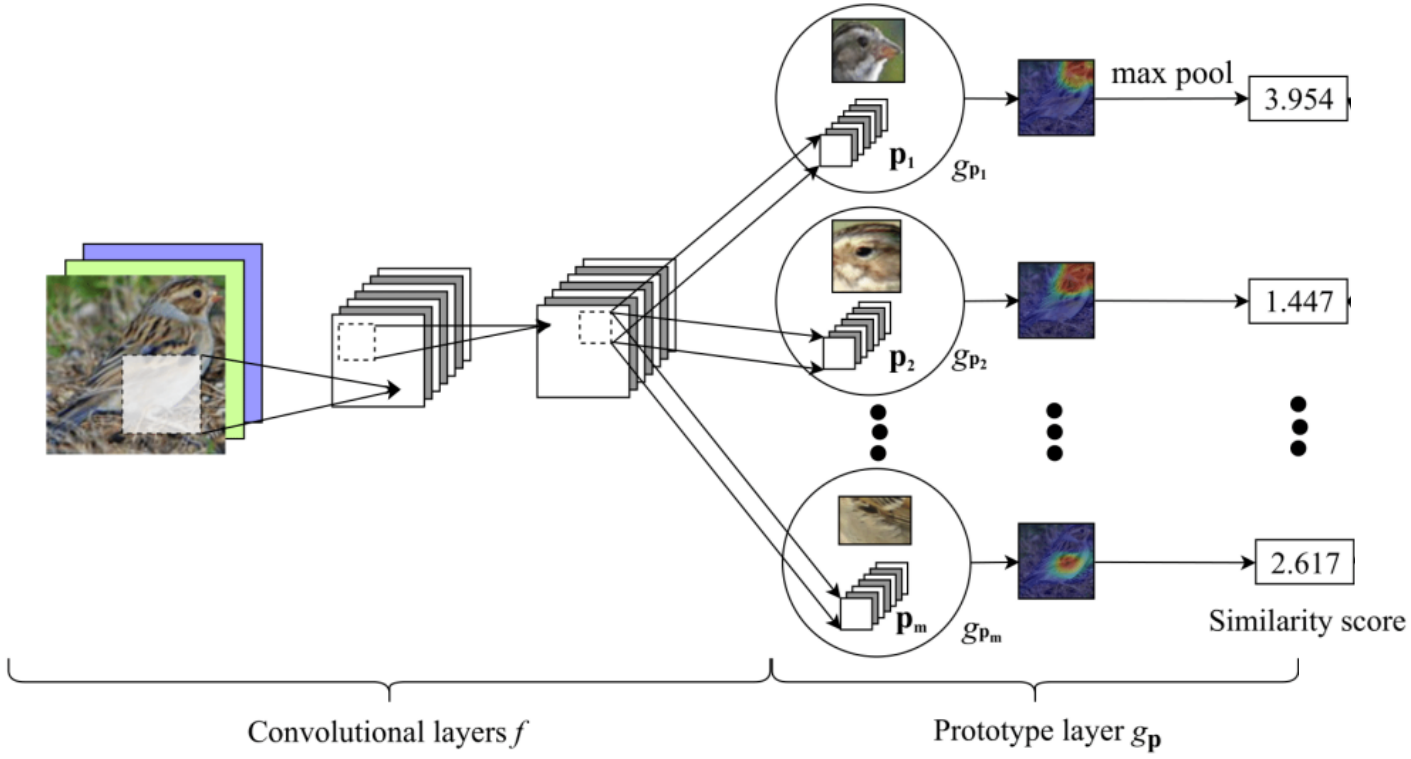


Figure 2: ProtoPNet architecture.

Prototypical part network (ProtoPNet)

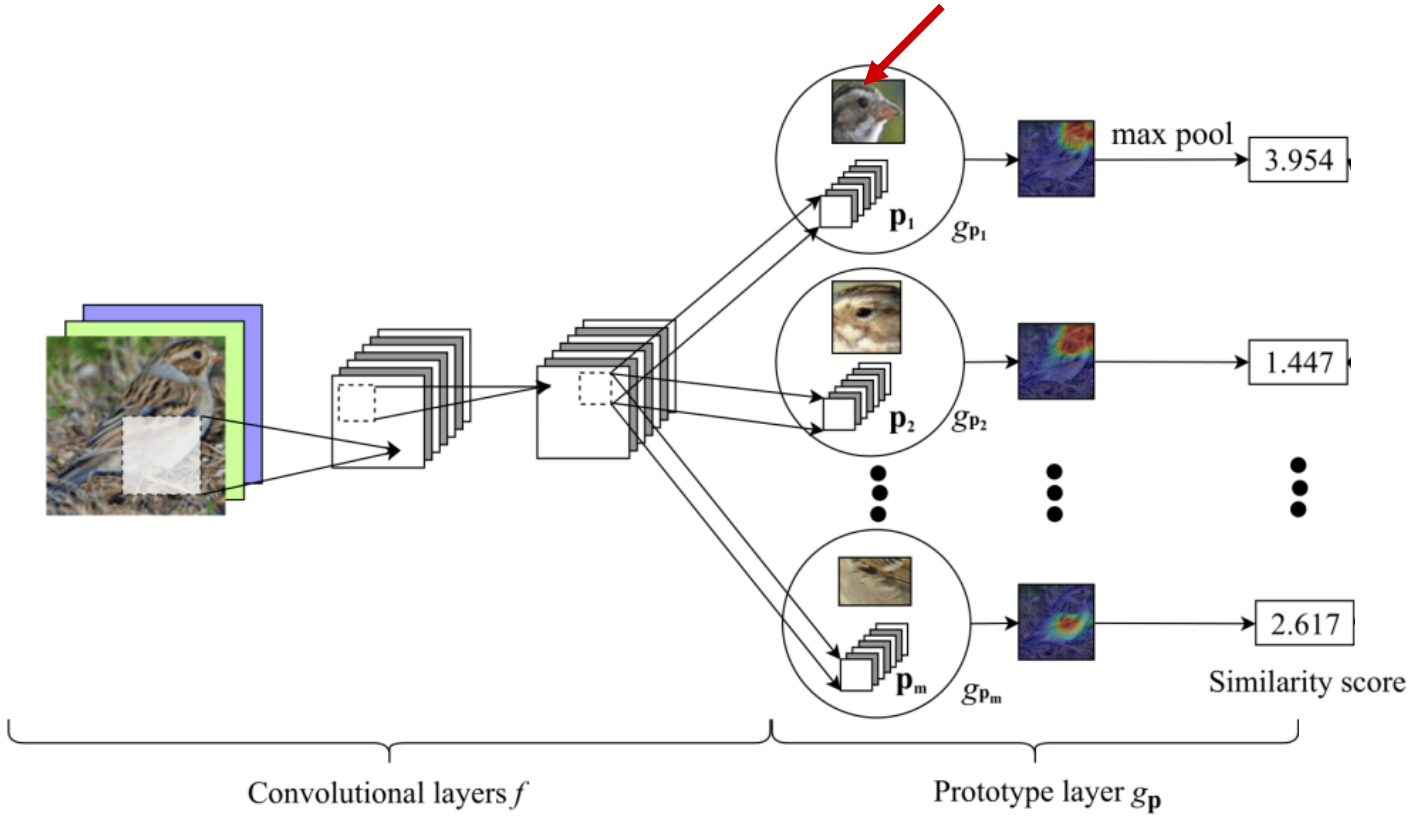


Figure 2: ProtoPNet architecture.

Prototypical part network (ProtoPNet)

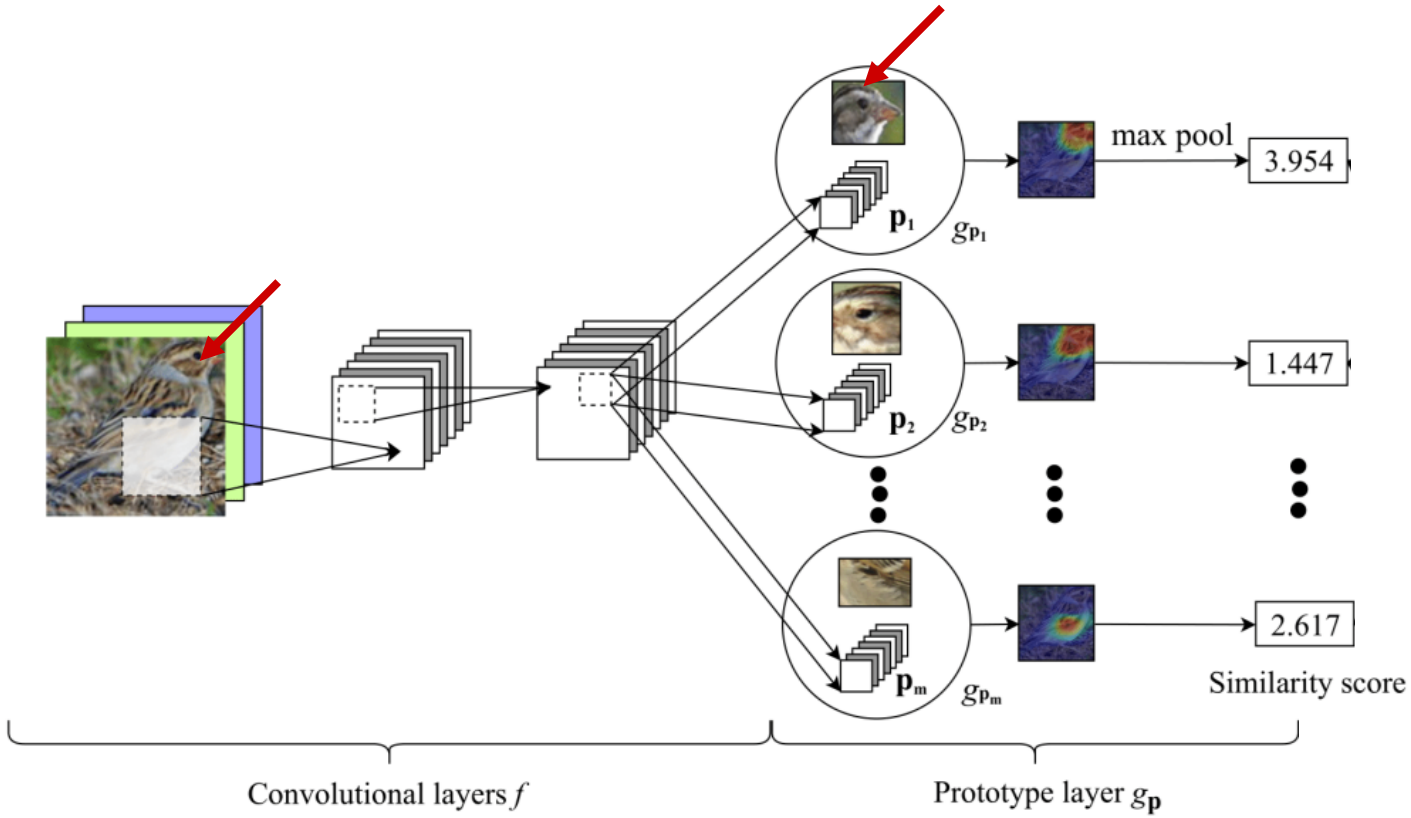


Figure 2: ProtoPNet architecture.

Prototypical part network (ProtoPNet)

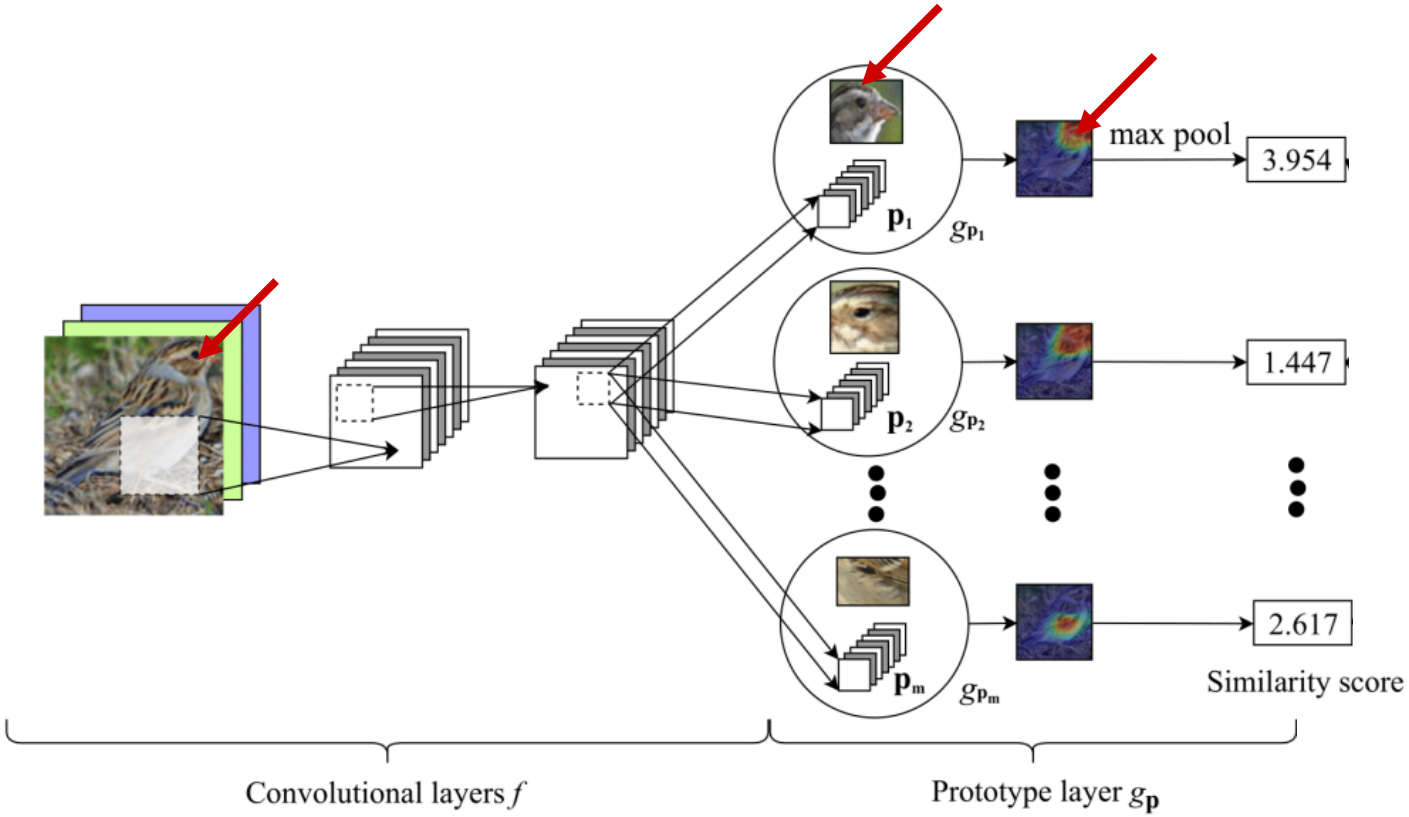


Figure 2: ProtoPNet architecture.

Prototypical part network (ProtoPNet)

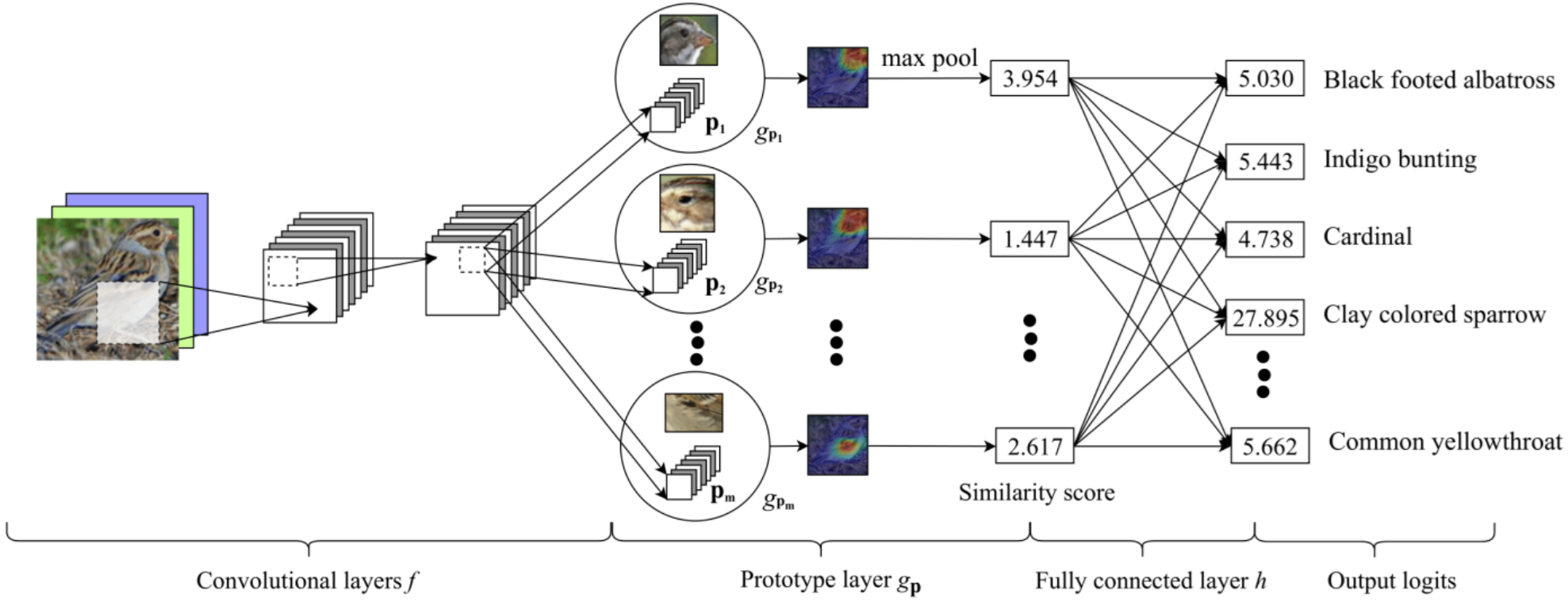


Figure 2: ProtoPNet architecture.

Prototypical part network (ProtoPNet)

Training:

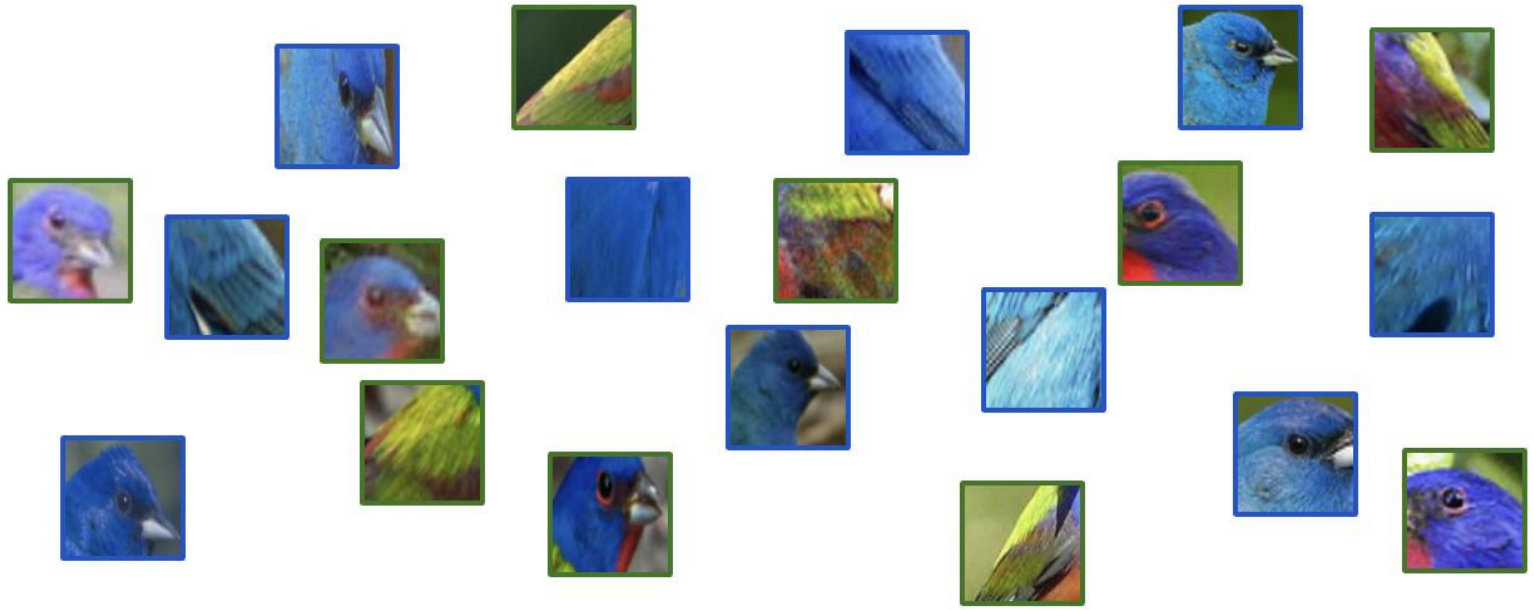
- **Stage 1**: Stochastic gradient descent (SGD) of layers before last layer
- **Stage 2**: Projection of prototypes
- **Stage 3**: Convex optimization of last layer

Prototypical part network (ProtoPNet)

- **Stage 1**: SGD of layers before last layer

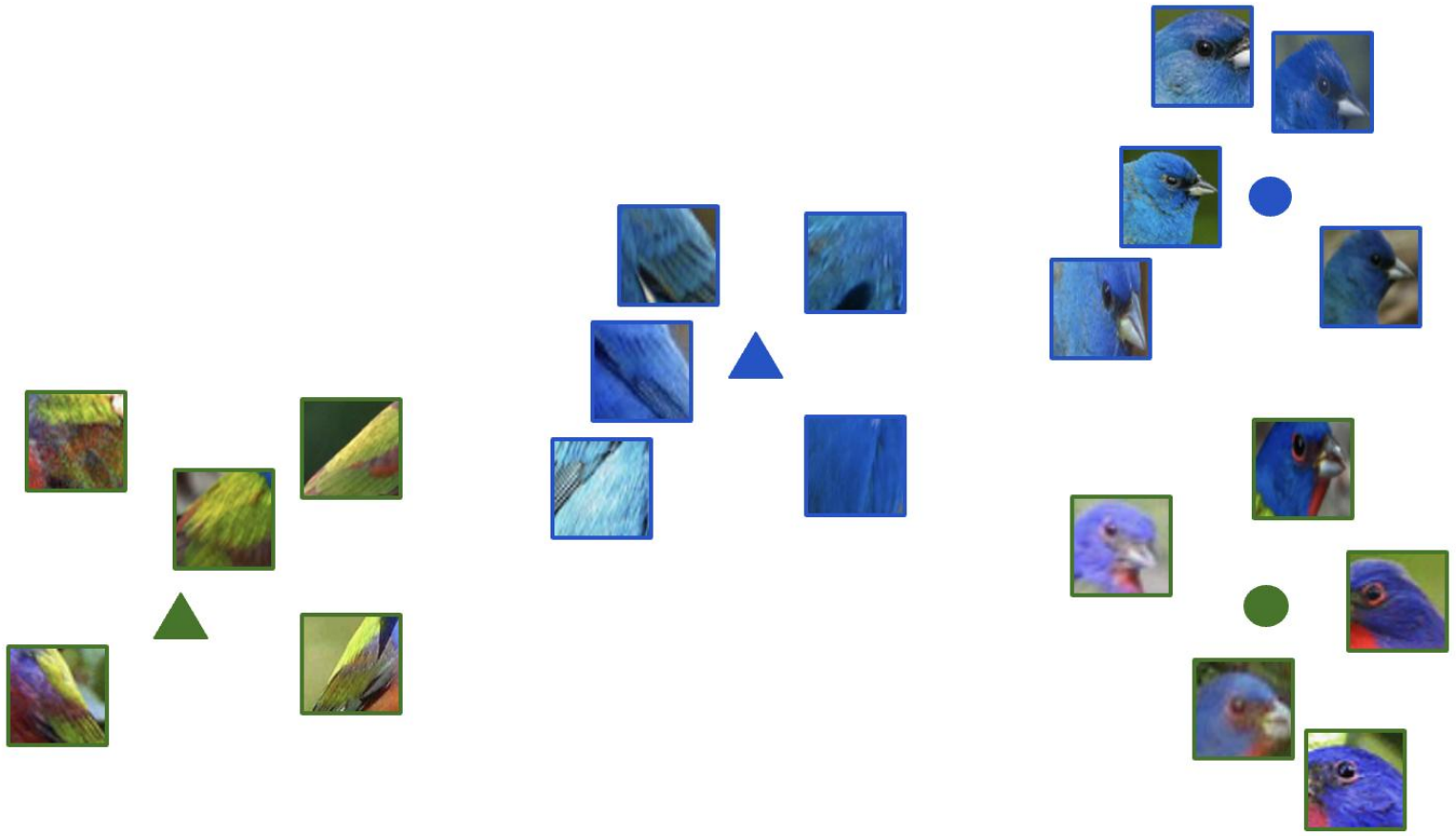
$$\min_{\mathbf{P}, w_{\text{conv}}} \frac{1}{n} \sum_{i=1}^{n} \text{CrsEnt}(h \circ g_{\mathbf{p}} \circ f(\mathbf{x_i}), \mathbf{y_i}) + \lambda_1 \text{Clst} + \lambda_2 \text{Sep}, \quad \text{where Clst and Sep are defined by}$$

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



Prototypical part network (ProtoPNet)

- **Stage 1**: SGD of layers before last layer

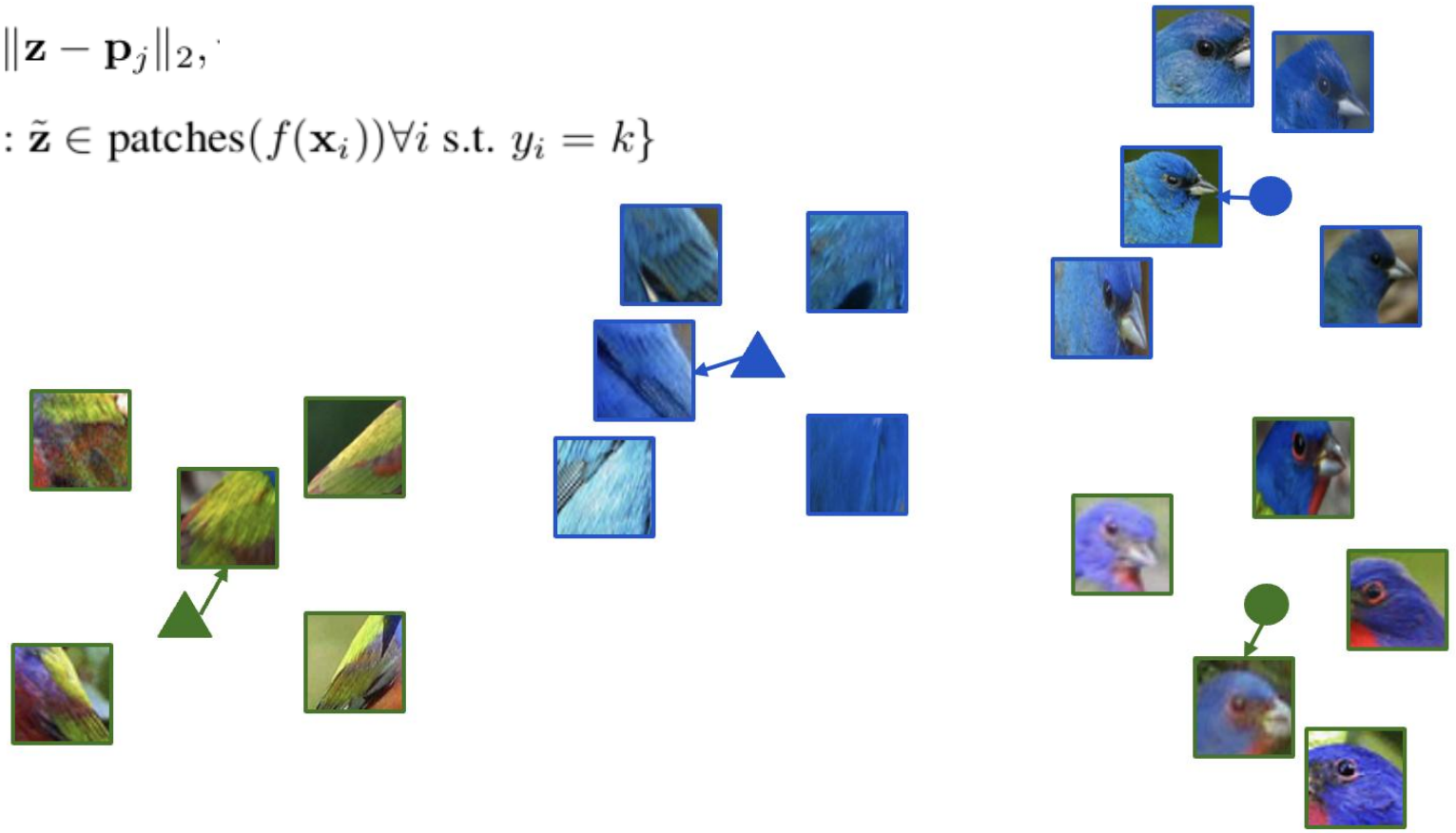


Prototypical part network (ProtoPNet)

- **Stage 2**: Projection of prototypes

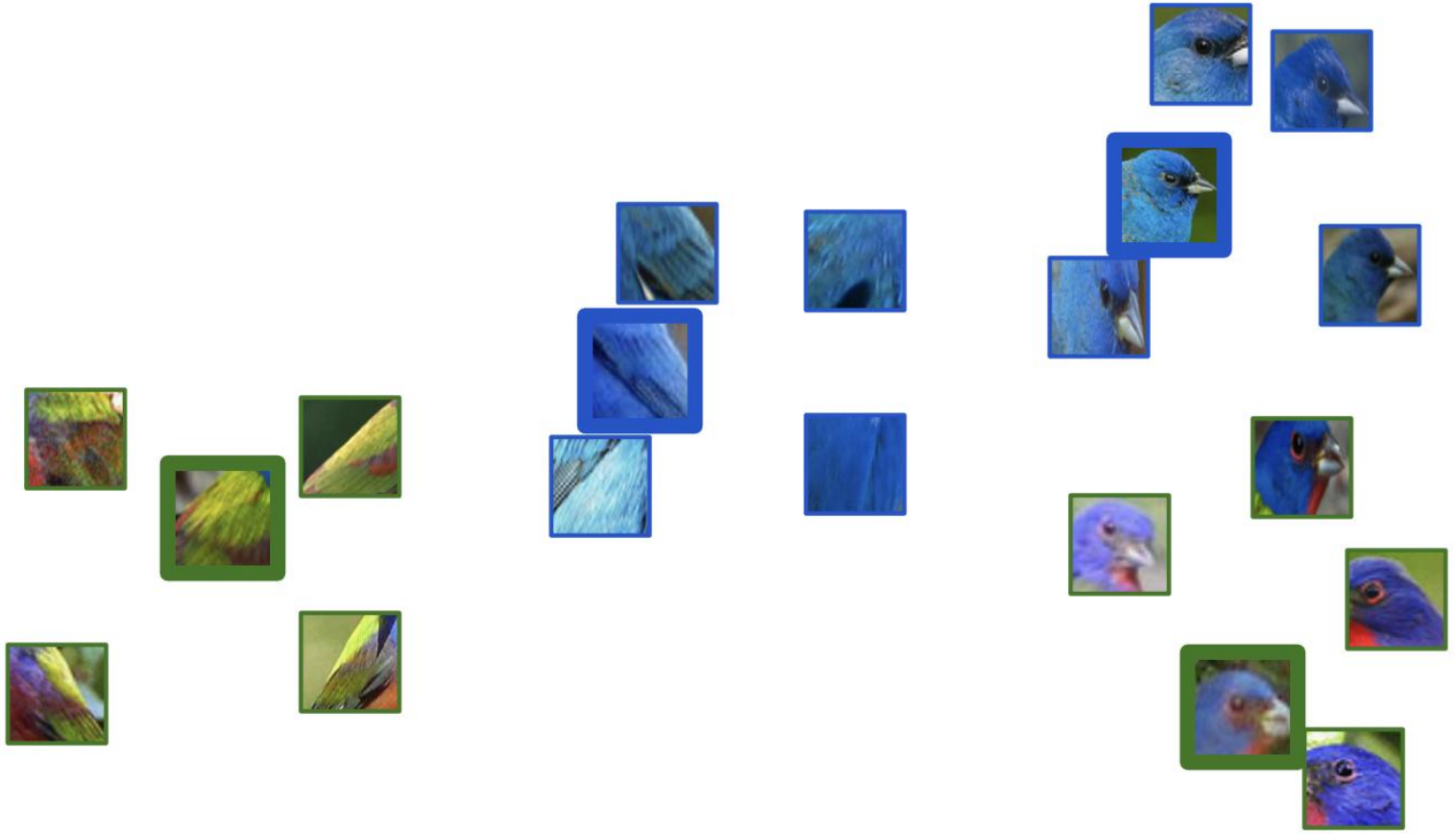
$$\mathbf{p}_j \leftarrow \arg\min_{\mathbf{z} \in \mathcal{Z}_j} \|\mathbf{z} - \mathbf{p}_j\|_2,$$

where $\mathcal{Z}_j = \{\tilde{\mathbf{z}} : \tilde{\mathbf{z}} \in \text{patches}(f(\mathbf{x}_i)) \forall i \text{ s.t. } y_i = k\}$



Prototypical part network (ProtoPNet)

- **Stage 2**: Projection of prototypes



Prototypical part network (ProtoPNet)

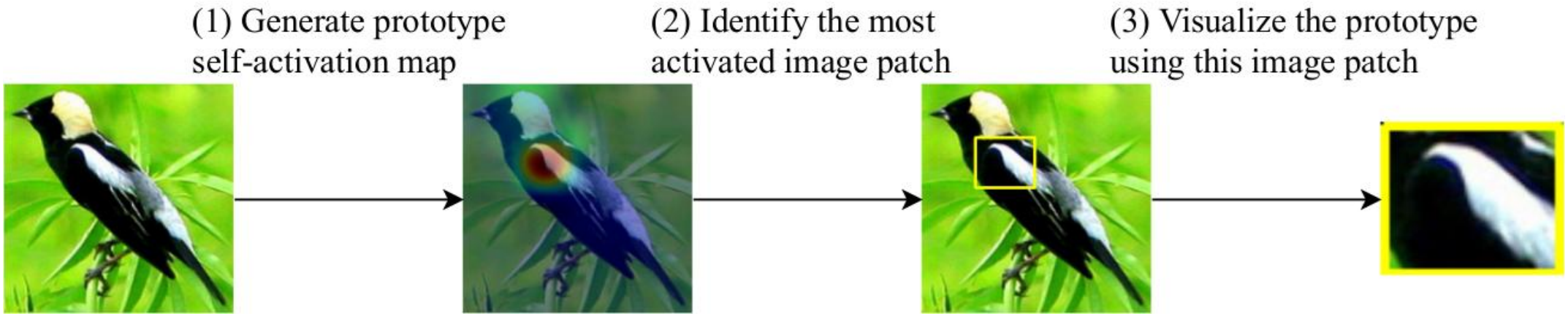
- **Stage 3**: Convex optimization of last layer

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

  - **No change in latent space**: layers up to and including the prototype layer are frozen

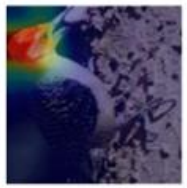
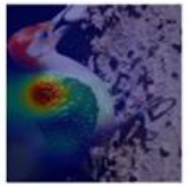
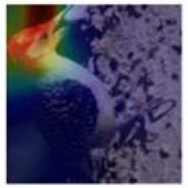
Prototypical part network (ProtoPNet)

- Prototype visualization



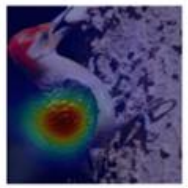
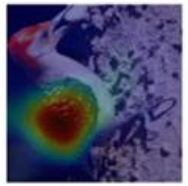
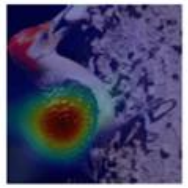
Why is this bird classfied as a red-bellied woodpecker?

Evidence for this bird being a red-bellied woodpecker:

Original image (box showing part that looks like prototype)	Prototype	Training image where prototype comes from	Activation map	Similarity score	Class connection	Points contributed
				6.499	1.180	7.669
				4.392	1.127	4.950
				3.890	1.108	4.310
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Total points to red-bellied woodpecker: 32.736

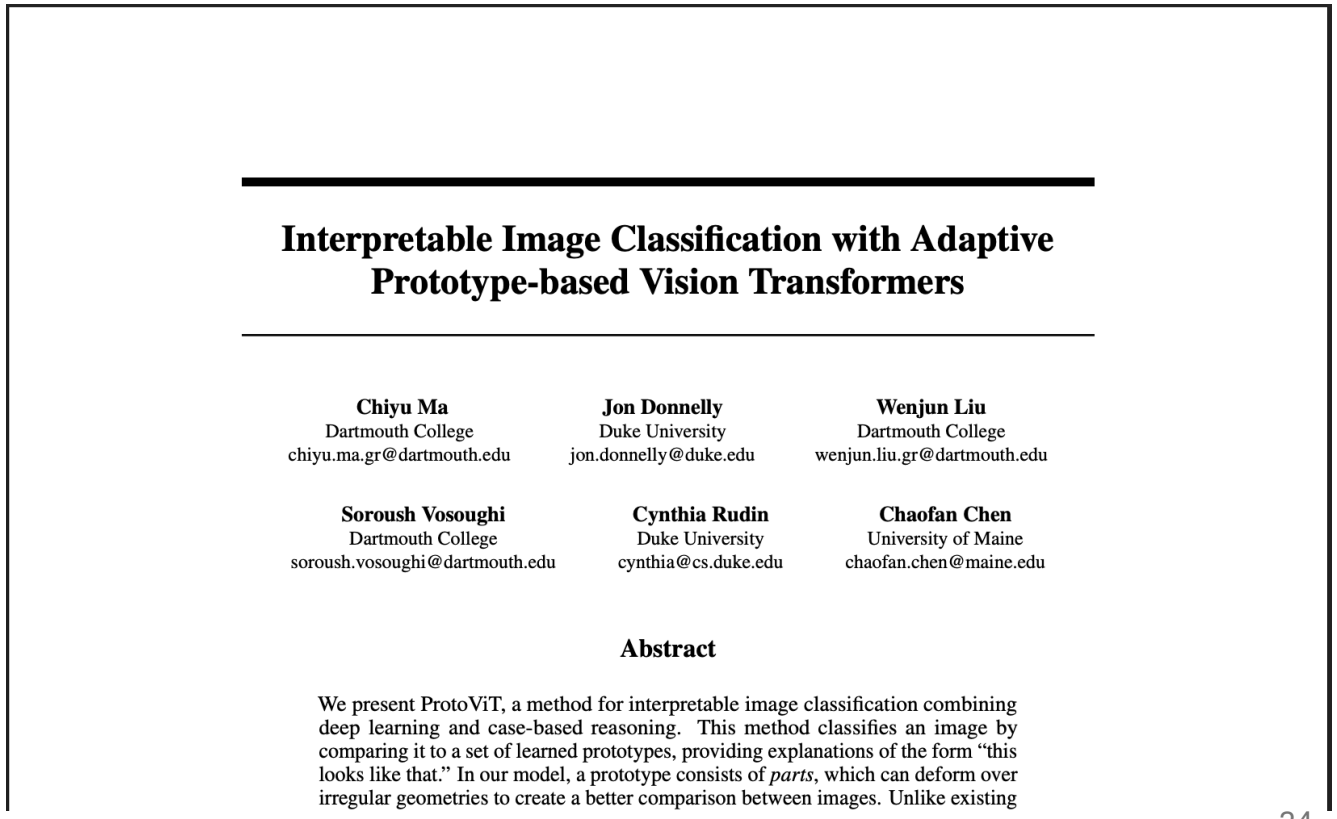
Evidence for this bird being a red-cockaded woodpecker:

Original image (box showing part that looks like prototype)	Prototype	Training image where prototype comes from	Activation map	Similarity score	Class connection	Points contributed
				2.452	1.046	2.565
				2.125	1.091	2.318
				1.945	1.069	2.079
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Total points to red-cockaded woodpecker: 16.886

ProtoViT

- “Interpretable Image Classification with Adaptive Prototype-based Vision Transformers” (NeurIPS, 2024)



Interpretable Image Classification with Adaptive Prototype-based Vision Transformers

Chiyu Ma
Dartmouth College
chiyu.ma.gr@dartmouth.edu

Jon Donnelly
Duke University
jon.donnelly@duke.edu

Wenjun Liu
Dartmouth College
wenjun.liu.gr@dartmouth.edu

Soroush Vosoughi
Dartmouth College
soroush.vosoughi@dartmouth.edu

Cynthia Rudin
Duke University
cynthia@cs.duke.edu

Chaofan Chen
University of Maine
chaofan.chen@maine.edu

Abstract

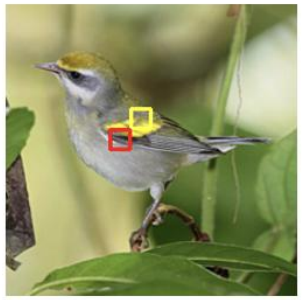
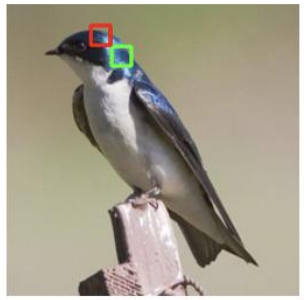
We present ProtoViT, a method for interpretable image classification combining deep learning and case-based reasoning. This method classifies an image by comparing it to a set of learned prototypes, providing explanations of the form “this looks like that.” In our model, a prototype consists of *parts*, which can deform over irregular geometries to create a better comparison between images. Unlike existing

ProtoViT

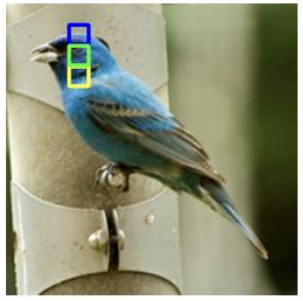
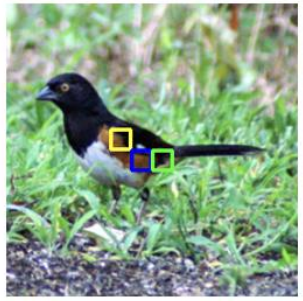
- Supports prototypes with *variable* numbers of tokens…

Prototypes with 1 token:


Prototypes with 2 tokens:

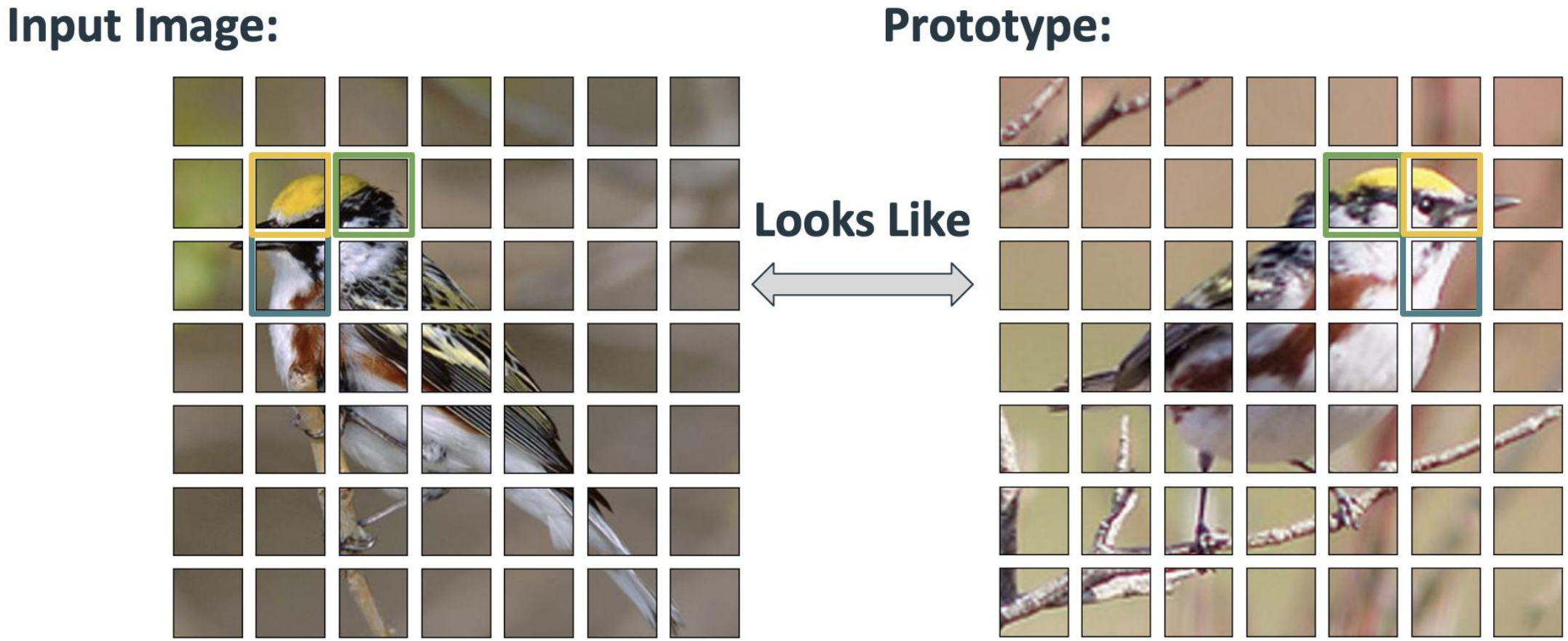
Prototypes with 3 tokens:

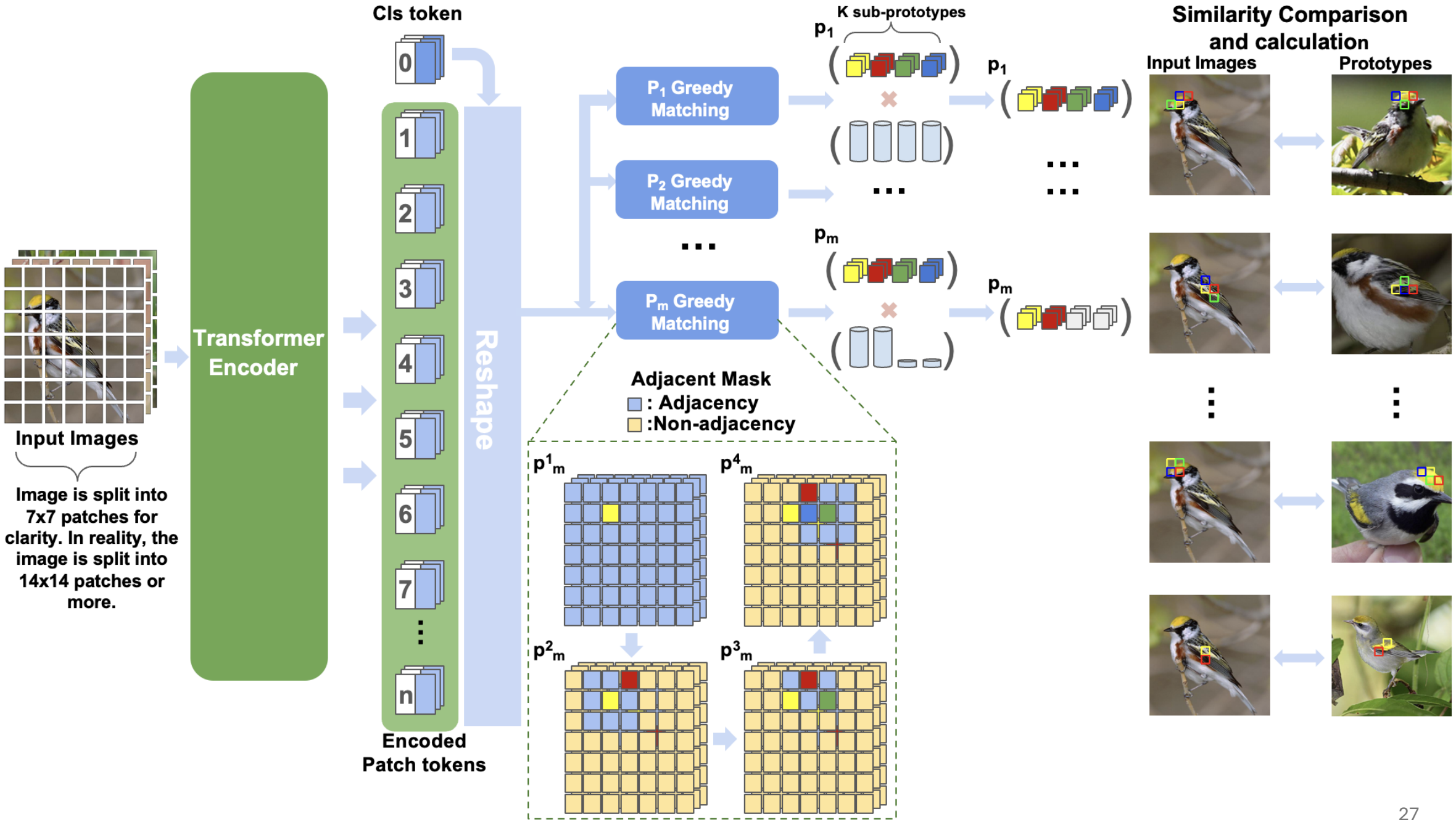



Prototypes with 4 tokens:


ProtoViT

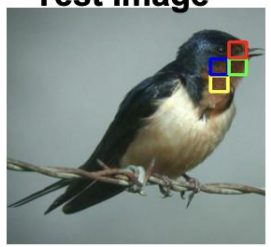
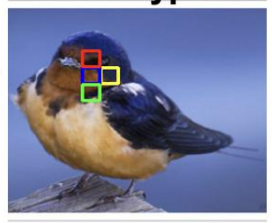
- ...that can deform (via *greedy matching with adjacency masking*)



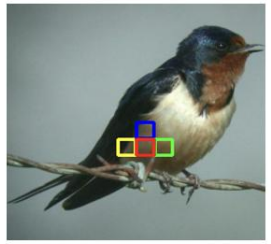

Cls token
K sub-prototypes
Similarity Comparison and calculation
Input Images
Prototypes
Transformer Encoder
Reshape
P1 Greedy Matching
P2 Greedy Matching
Pm Greedy Matching
Adjacent Mask
: Adjacency
:Non-adjacency
Input Images
Image is split into 7x7 patches for clarity. In reality, the image is split into 14x14 patches or more.
Encoded Patch tokens

Reasoning for this bird being a Barn Swallow:

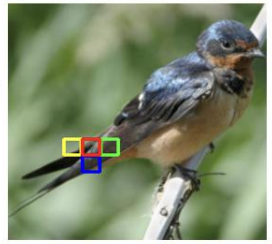
Test Image | Looks like | Prototypes | Reweighed Cumulative similarity | Last Layer weight | Total weight

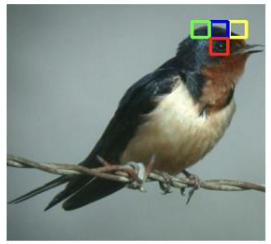
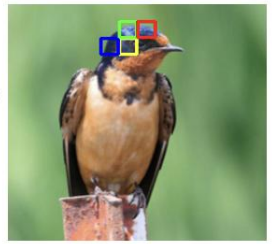
3.65 X 0.901 = 3.28




3.61 X 0.902 = 3.26

3.58 X 0.904 = 3.25

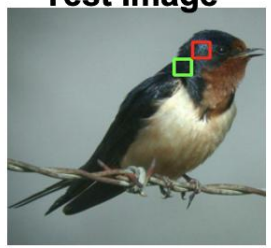
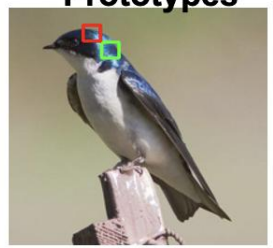
3.56 X 0.900 = 3.20

⋮ ⋮ ⋮

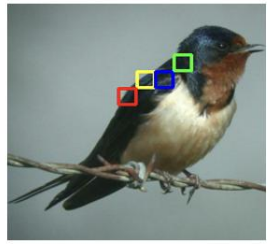
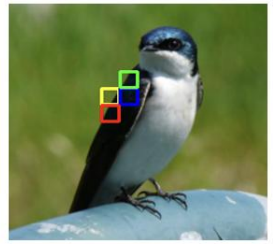
Total Points to Barn Swallow: 31.6

Reasoning for this bird being a Tree Swallow:

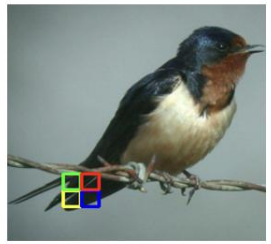
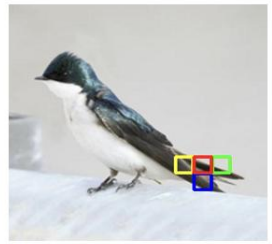
Test Image | Looks like | Prototypes | Reweighed Cumulative similarity | Last Layer weight | Total weight

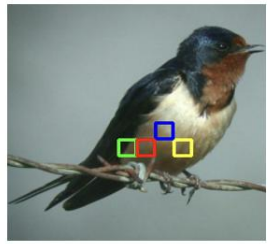
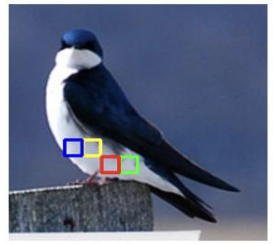
2.62 X 0.940 = 2.46

1.89 X 0.941 = 1.78

1.77 X 0.941 = 1.67

1.42 X 0.939 = 1.33

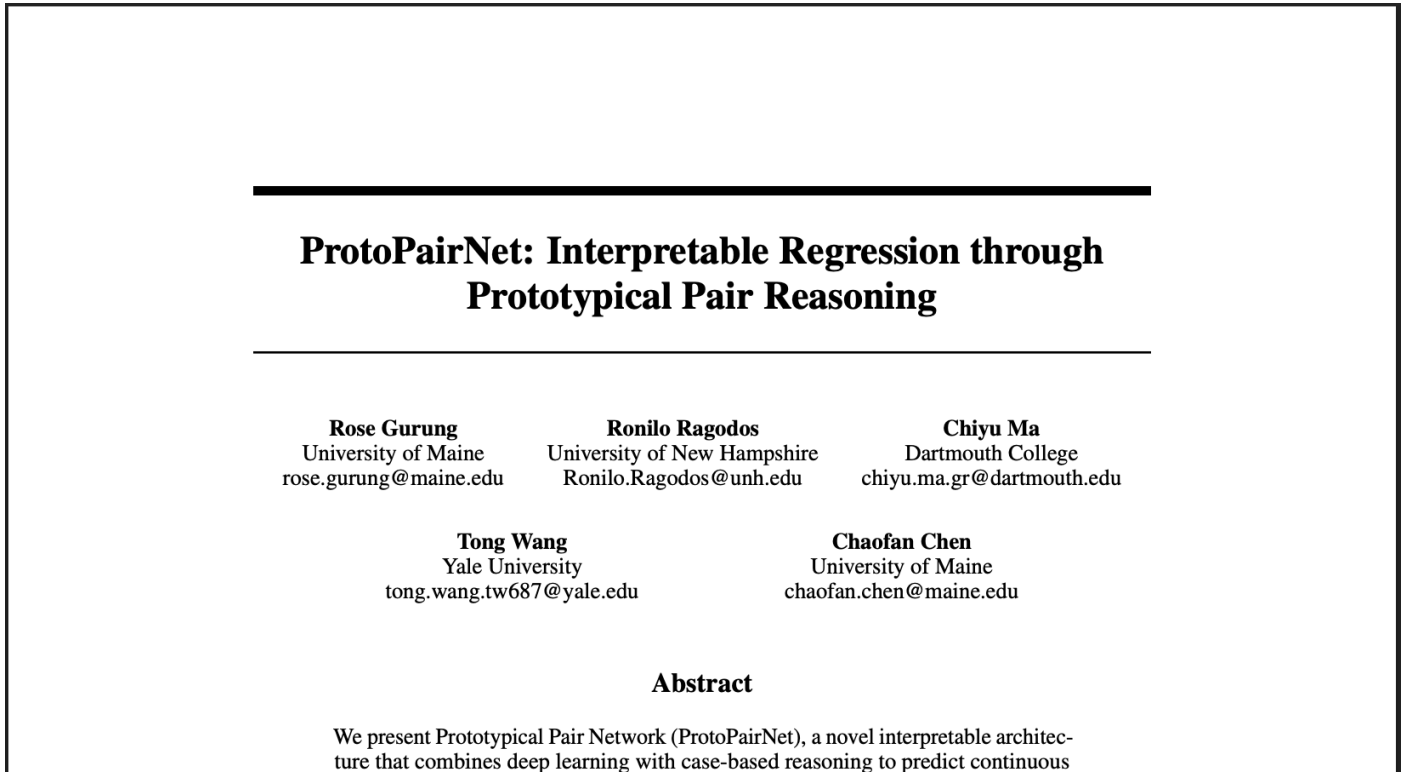
⋮ ⋮ ⋮

Total Points to Tree Swallow: 15.6

Arch.	Model	CUB Acc.[%]	Car Acc.[%]
Densenet-161 ~28.68M params	ProtoPNet (given in [9])	80.1 ± 0.3	89.5 ± 0.2
	Def. ProtoPNet(2x2) (given in [10])	80.9 ±0.22	88.7 ± 0.3
	ProtoPool (given in [13])	80.3 ± 0.3	90.0 ± 0.3
	TesNet (given in [14])	**81.5 ± 0.3**	**92.6 ± 0.3**
DeiT-Tiny ~5M params	Base (given in [23])	80.57	86.21
	ViT-Net (given in [23])	81.98	88.41
	ProtoPFormer (given in [23])	82.26	88.48
	ProtoViT(K=4, r=1) (ours)	**82.92 ± 0.5**	**89.02 ± 0.1**
DeiT-Small ~22M params	Baseline (given in [23])	84.28	90.06
	ViT-Net (given in [23])	84.26	91.34
	ProtoPFormer (given in [23])	84.85	90.86
	ProtoViT(K=4, r=1) (ours)	**85.37 ± 0.13**	**91.84 ± 0.3**
CaiT-XXS 24 ~11.9M params	Baseline (given in [23])	83.95	90.19
	ViT-Net (given in [23])	84.51	91.54
	ProtoPFormer (given in [23])	84.79	91.04
	ProtoViT(K=4, r=1) (ours)	**85.82 ± 0.15**	**92.40 ± 0.1**

ProtoPairNet

- “ProtoPairNet: Interpretable Regression through Prototypical Pair Reasoning” (NeurIPS, 2025)



ProtoPairNet: Interpretable Regression through Prototypical Pair Reasoning

Rose Gurung
University of Maine
rose.gurung@maine.edu

Ronilo Ragodos
University of New Hampshire
Ronilo.Ragodos@unh.edu

Chiyu Ma
Dartmouth College
chiyu.ma.gr@dartmouth.edu

Tong Wang
Yale University
tong.wang.tw687@yale.edu

Chaofan Chen
University of Maine
chaofan.chen@maine.edu

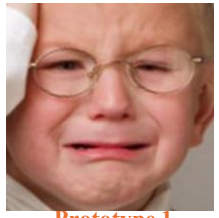
Abstract

We present Prototypical Pair Network (ProtoPairNet), a novel interpretable architec-
ture that combines deep learning with case-based reasoning to predict continuous

ProtoPairNet

Reasoning with one-to-one comparison:

Input Image
Original age 26

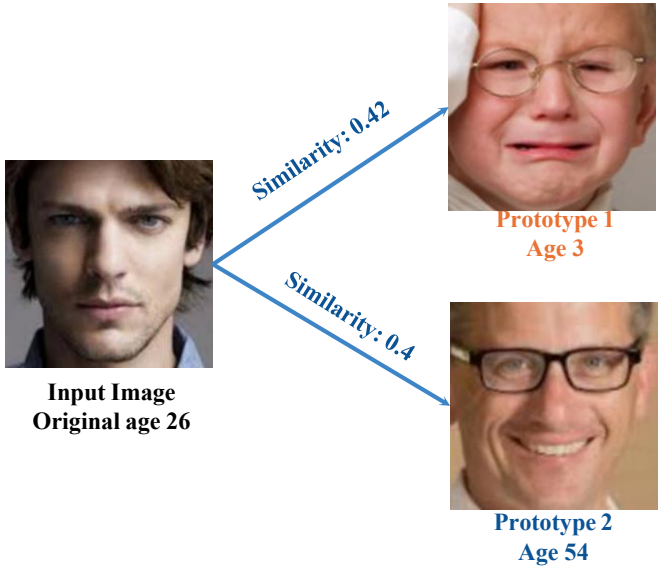


Prototype 1
Age 3

Prototype 2
Age 54

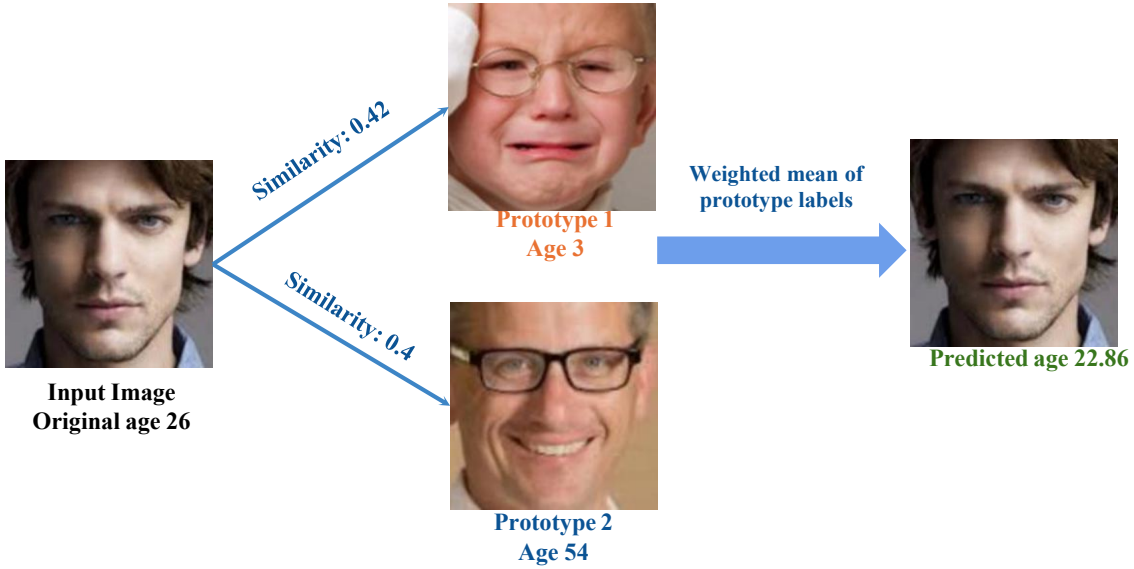
ProtoPairNet

Reasoning with one-to-one comparison:



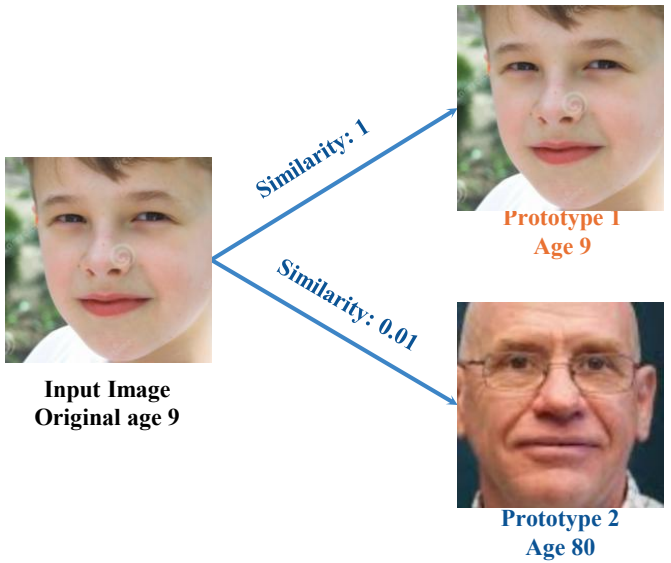
ProtoPairNet

Reasoning with one-to-one comparison:



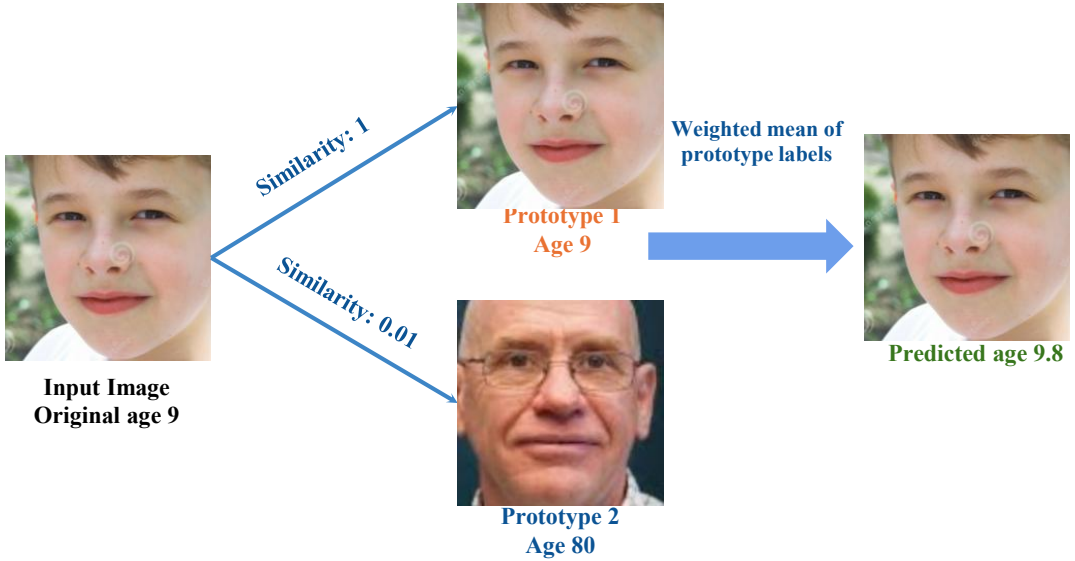
ProtoPairNet

Reasoning with one-to-one comparison:



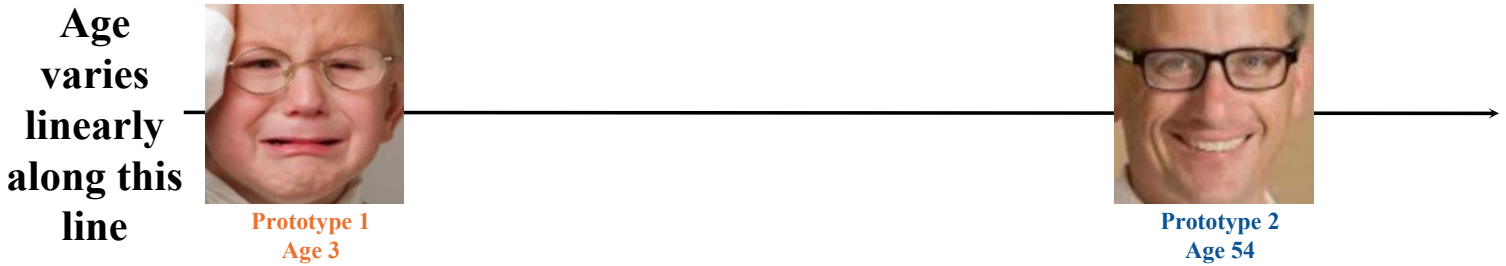
ProtoPairNet

Reasoning with one-to-one comparison:



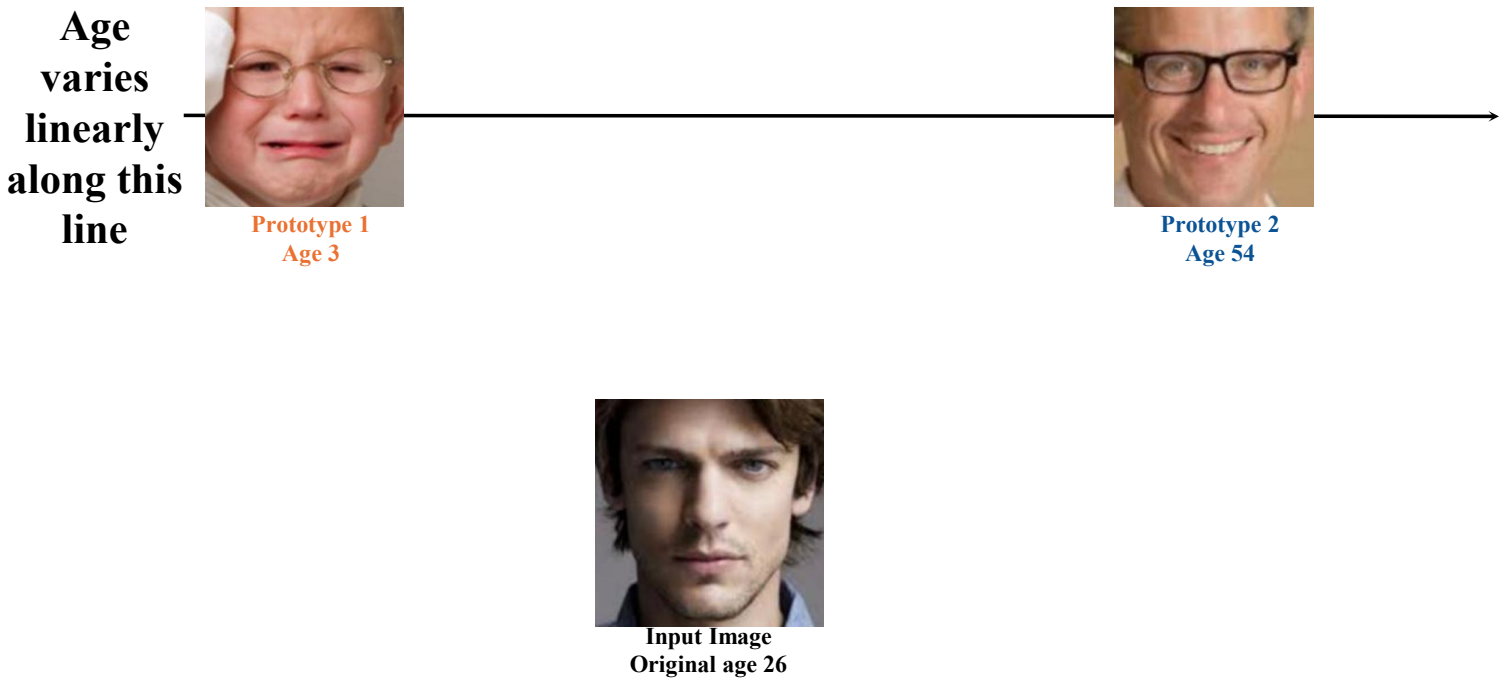
ProtoPairNet

Reasoning with prototypical pairs:



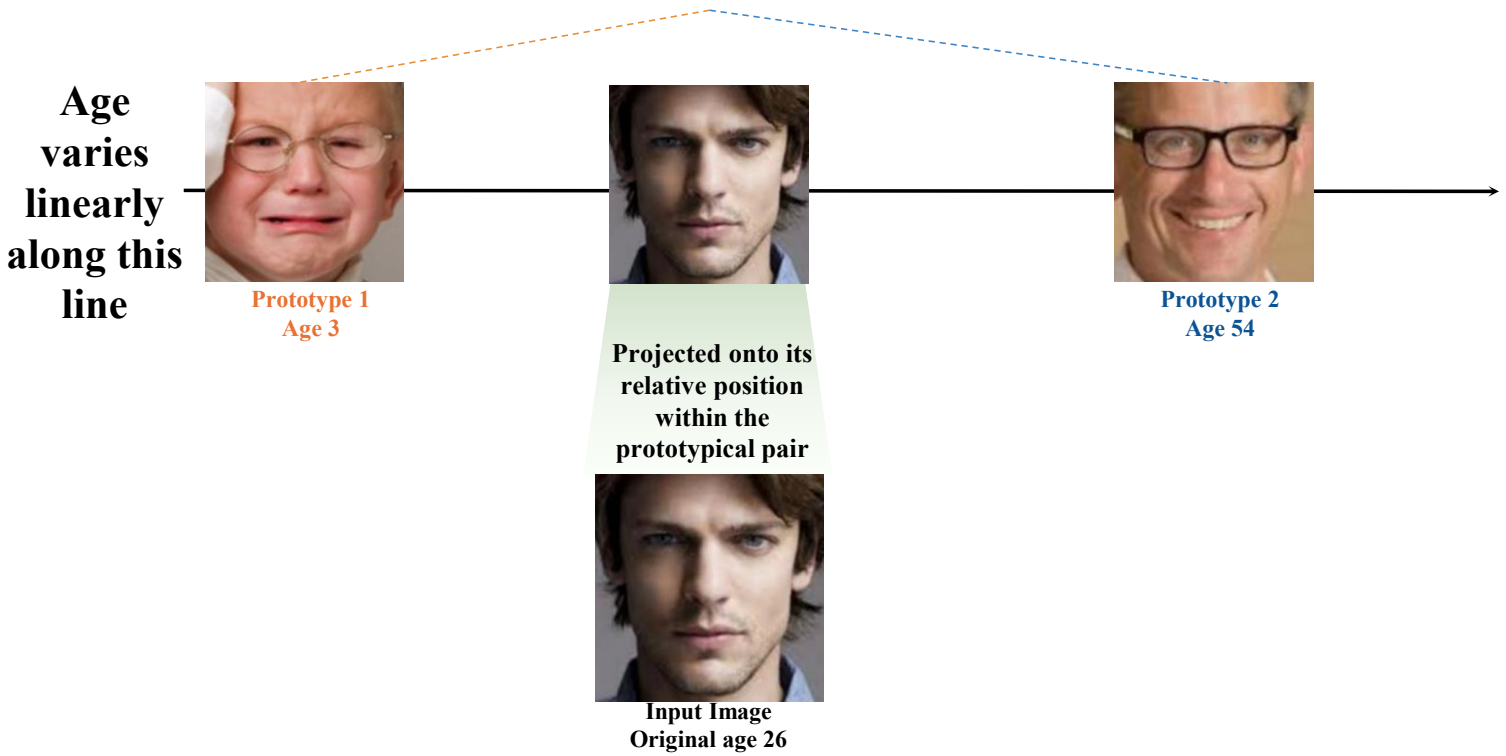
ProtoPairNet

Reasoning with prototypical pairs:



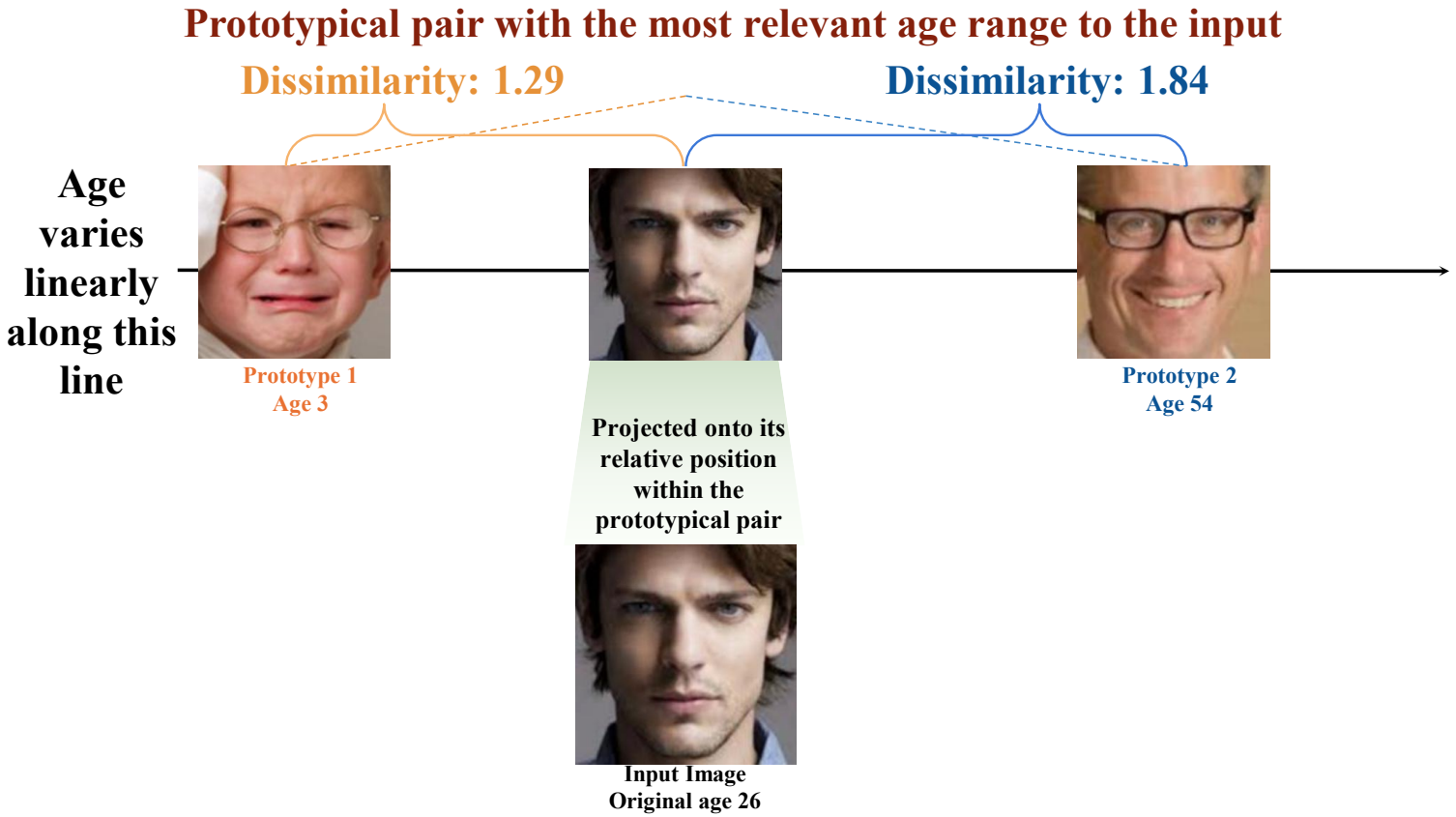
ProtoPairNet

Reasoning with prototypical pairs:

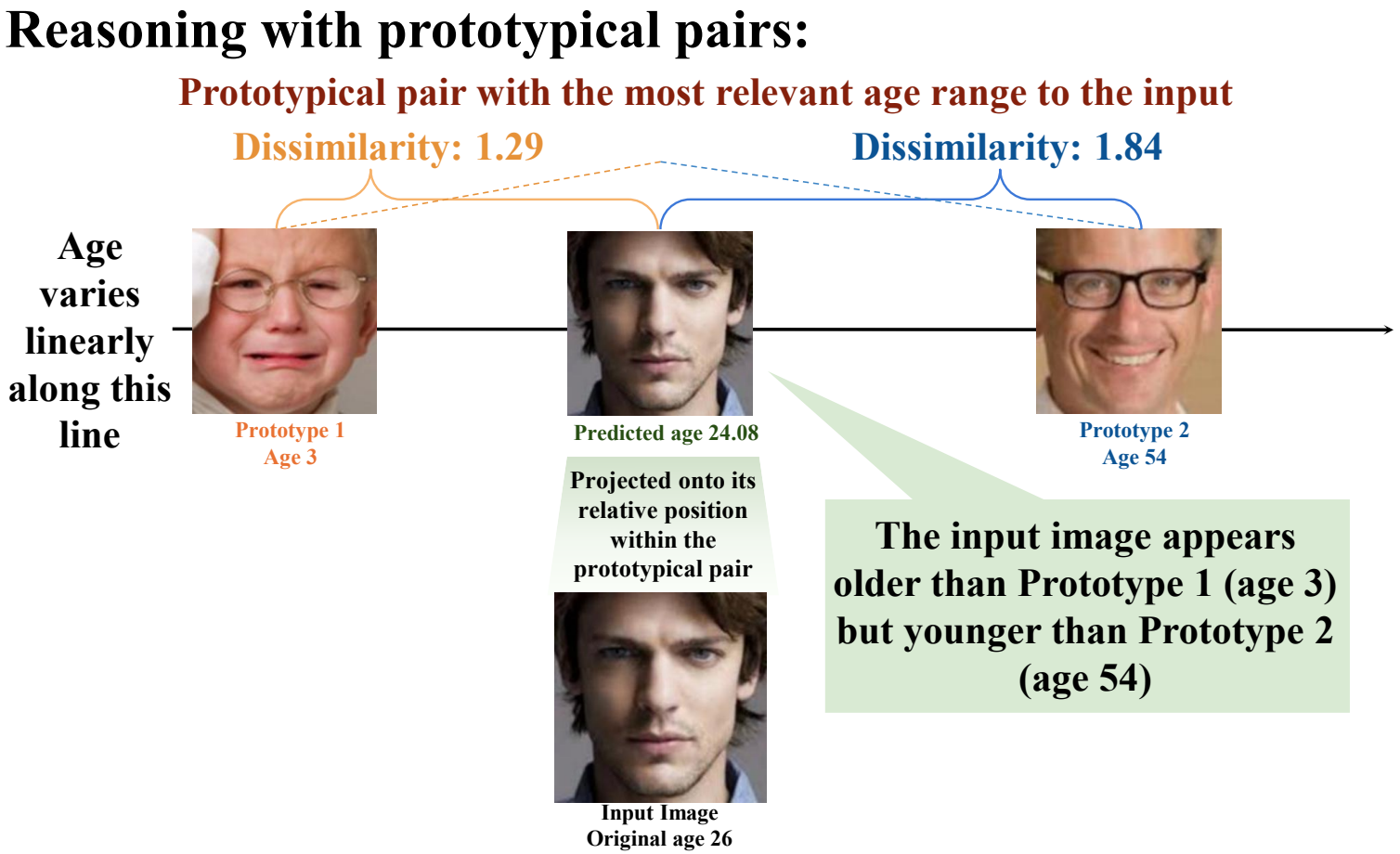


ProtoPairNet

Reasoning with prototypical pairs:



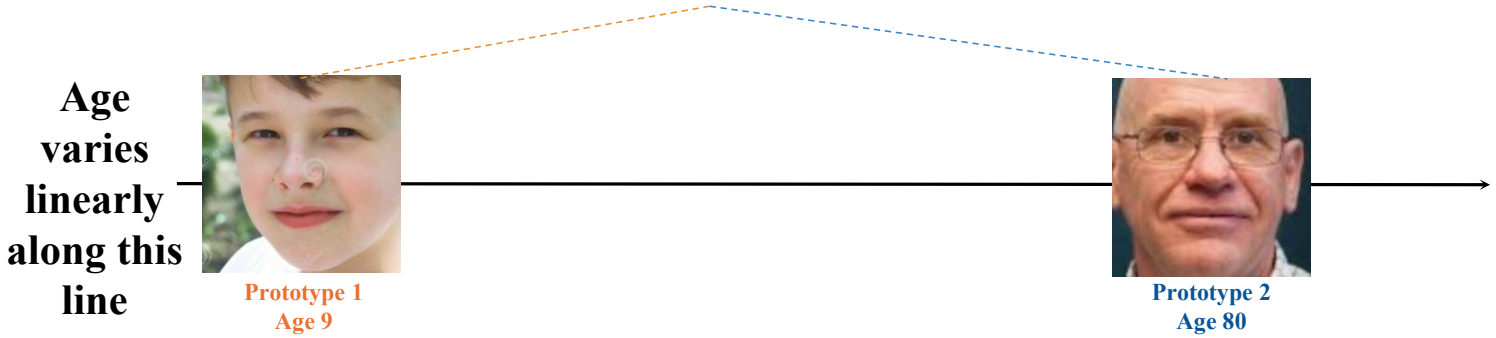
ProtoPairNet



ProtoPairNet

Reasoning with prototypical pairs:

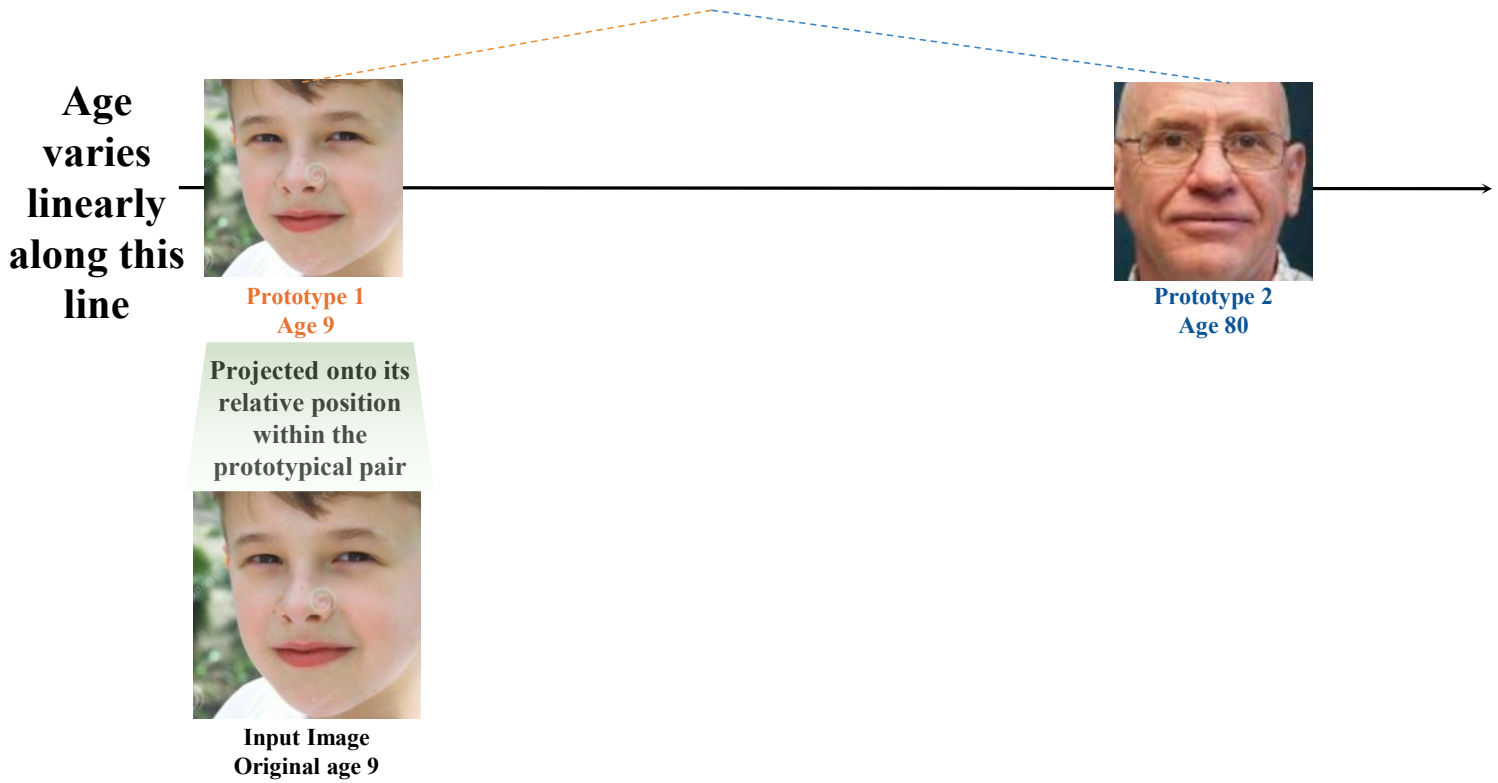
Prototypical pair with the most relevant age range to the input



Input Image
Original age 9

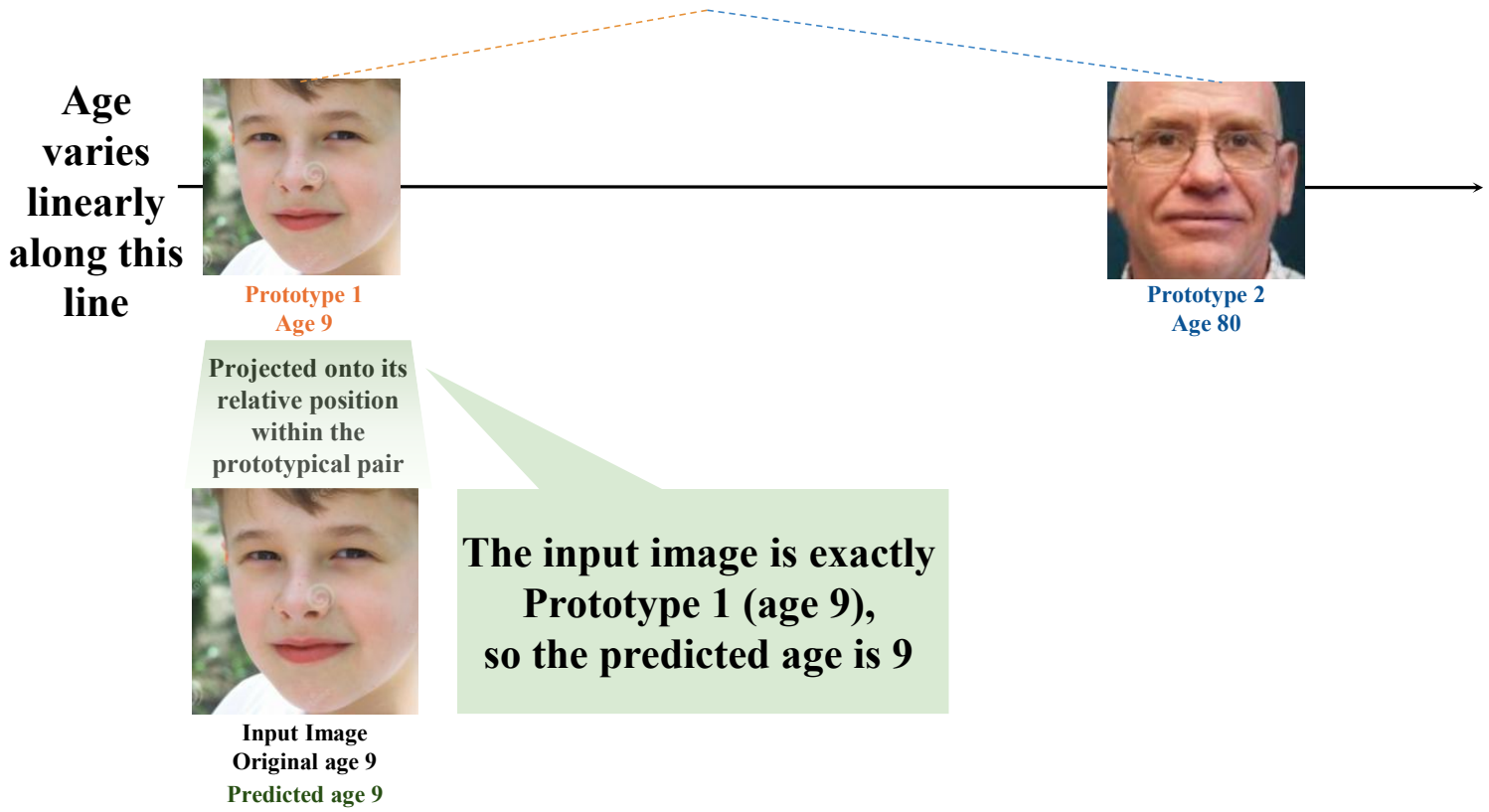
ProtoPairNet

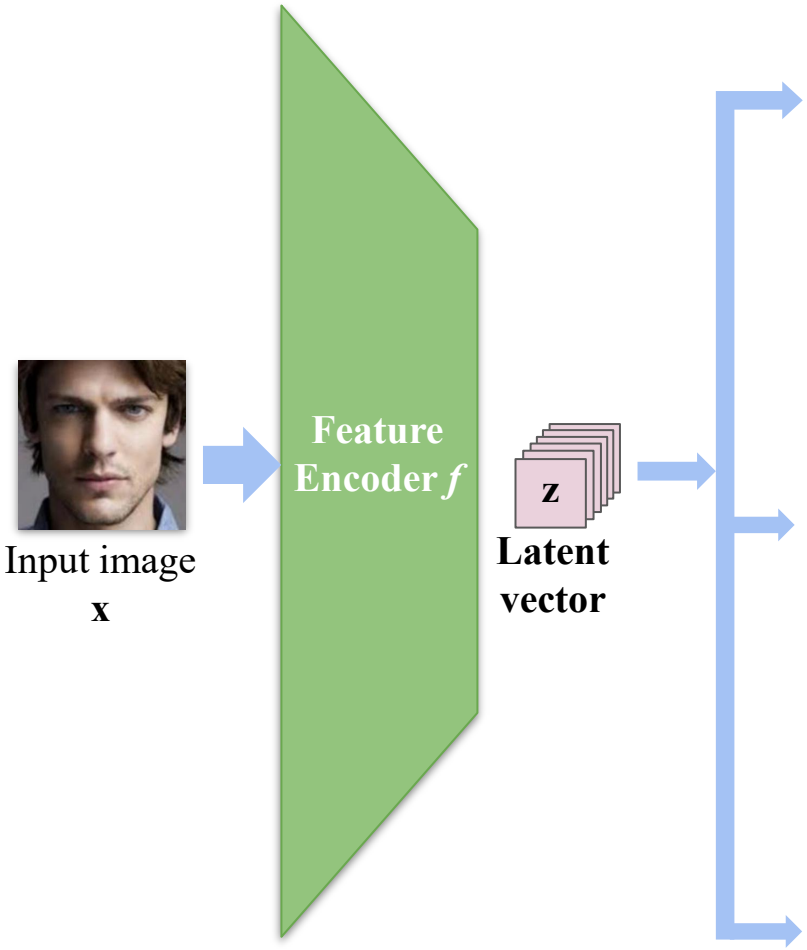
Reasoning with prototypical pairs:



ProtoPairNet

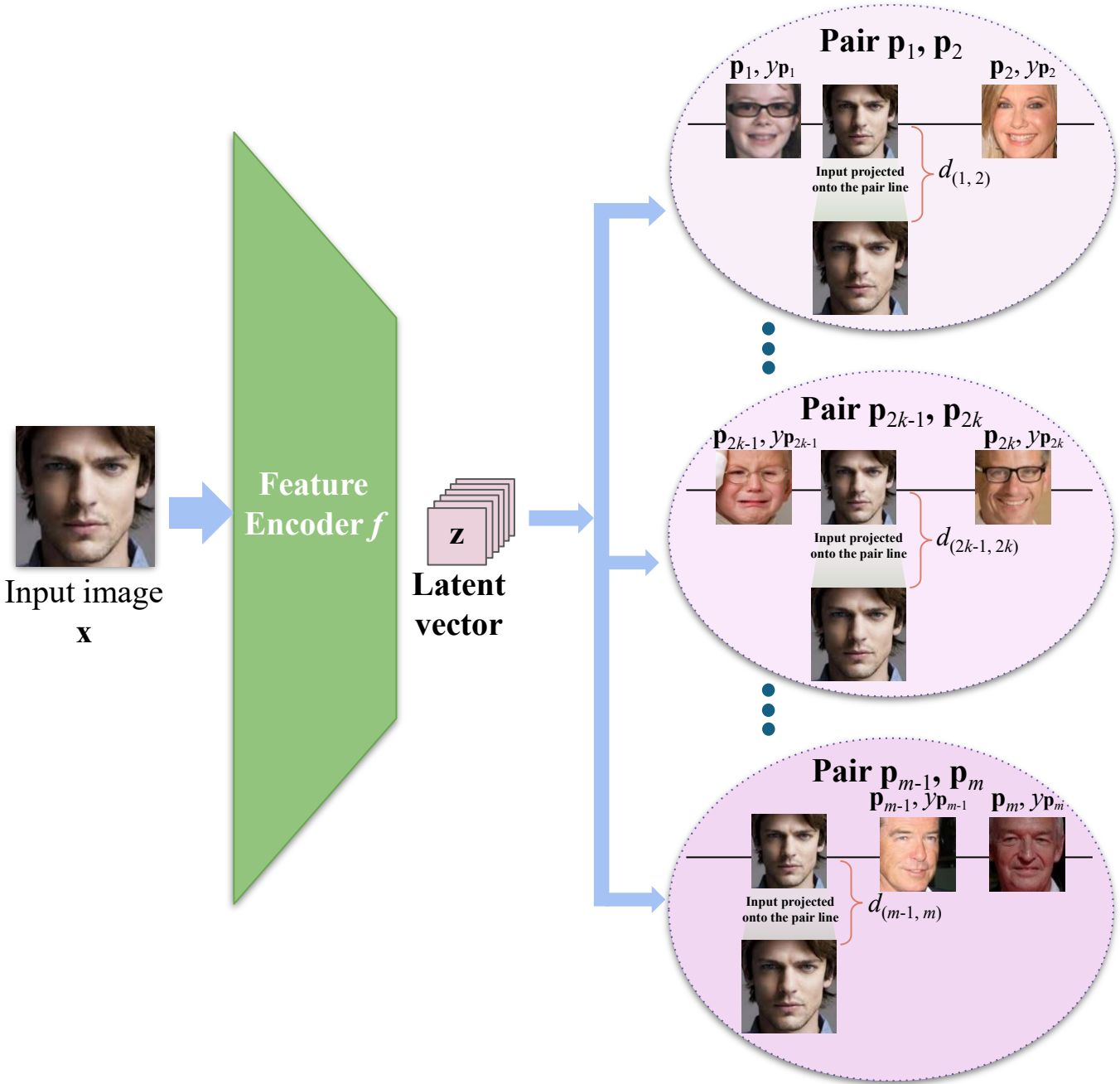
Reasoning with prototypical pairs:




Feature
Encoder *f*
z
Latent
vector
Input image
x

Prototypical pairs and corresponding labels:

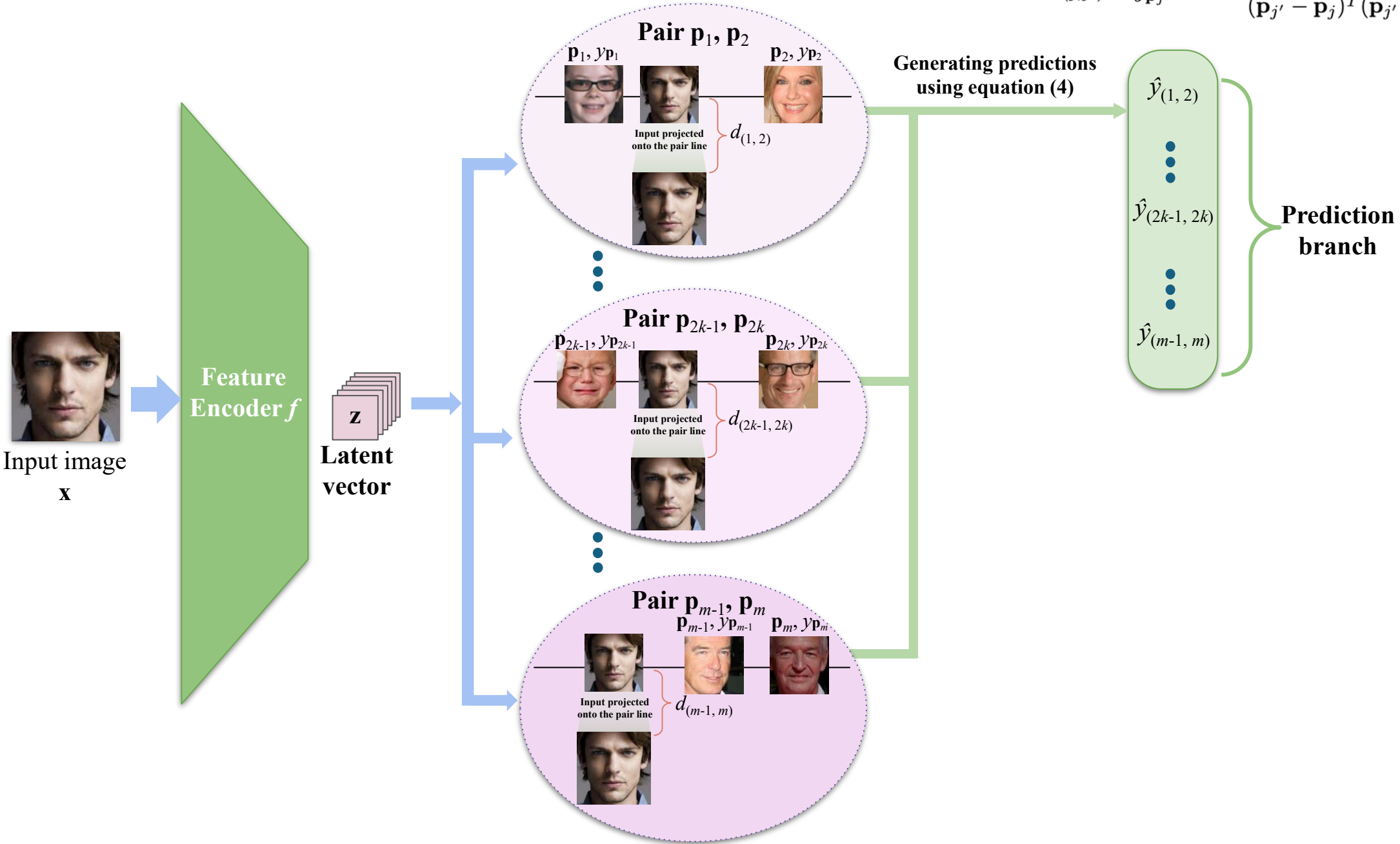
$(\mathbf{p}_1, \mathbf{p}_2), \ldots, (\mathbf{p}_{2k-1}, \mathbf{p}_{2k}), \ldots, (\mathbf{p}_{m-1}, \mathbf{p}_m)$, and $(y\mathbf{p}_1, y\mathbf{p}_2), \ldots, (y\mathbf{p}_{2k-1}, y\mathbf{p}_{2k}), \ldots, (y\mathbf{p}_{m-1}, y\mathbf{p}_m)$



Prototypical pairs and corresponding labels:

$(\mathbf{p}_1, \mathbf{p}_2), \ldots, (\mathbf{p}_{2k-1}, \mathbf{p}_{2k}), \ldots, (\mathbf{p}_{m-1}, \mathbf{p}_m)$, and $(y\mathbf{p}_1, y\mathbf{p}_2), \ldots, (y\mathbf{p}_{2k-1}, y\mathbf{p}_{2k}), \ldots, (y\mathbf{p}_{m-1}, y\mathbf{p}_m)$

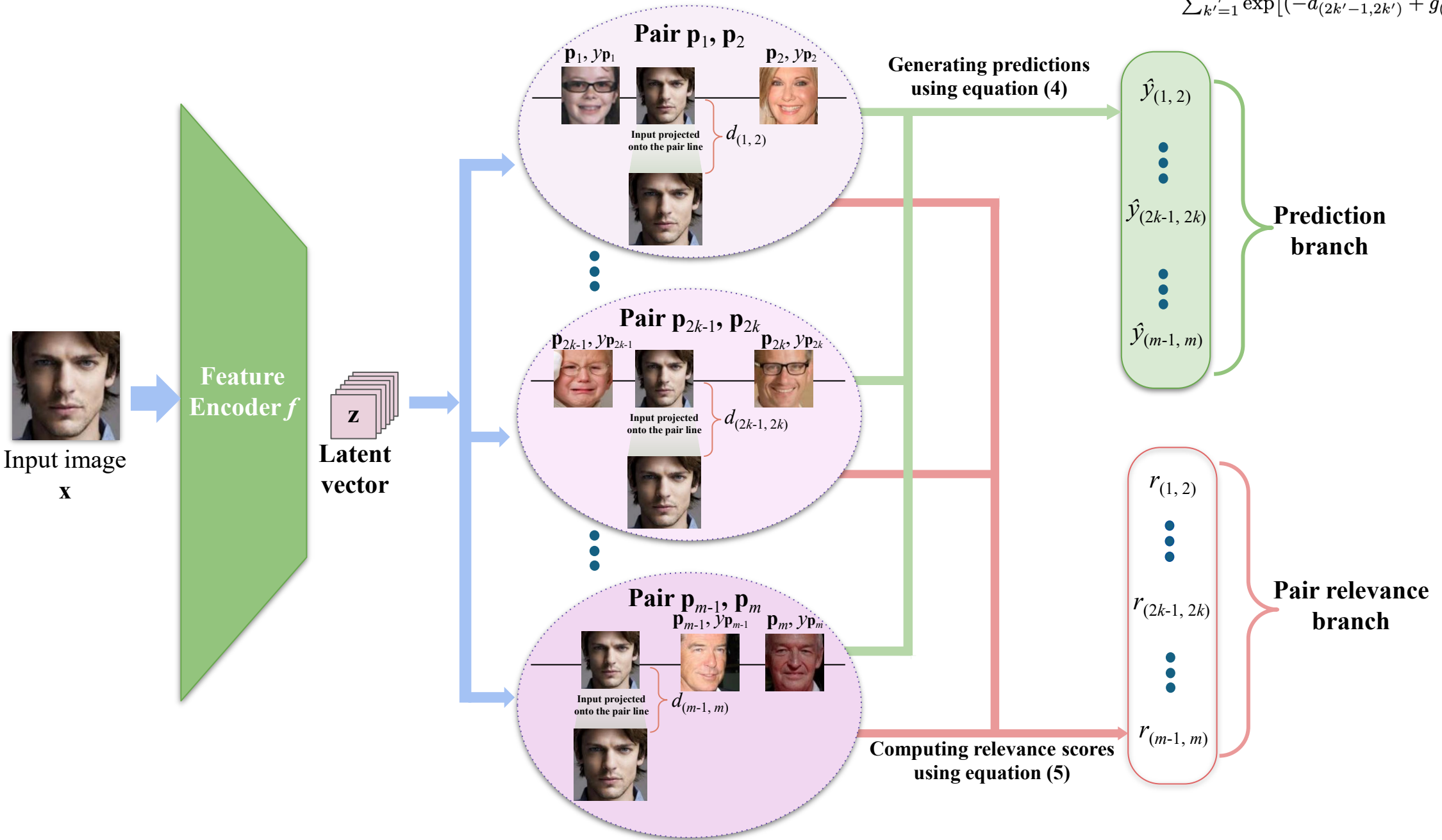
$$\hat{y}_{(j,j')} = y_{\mathbf{p}_j} + \frac{[(\mathbf{z} - \mathbf{p}_j)^T(\mathbf{p}_{j'} - \mathbf{p}_j)]\,(y_{\mathbf{p}_{j'}} - y_{\mathbf{p}_j})}{(\mathbf{p}_{j'} - \mathbf{p}_j)^T(\mathbf{p}_{j'} - \mathbf{p}_j)} \quad (4)$$



Prototypical pairs and corresponding labels:

$(\mathbf{p}_1, \mathbf{p}_2), \ldots, (\mathbf{p}_{2k-1}, \mathbf{p}_{2k}), \ldots, (\mathbf{p}_{m-1}, \mathbf{p}_m)$, and $(y\mathbf{p}_1, y\mathbf{p}_2), \ldots, (y\mathbf{p}_{2k-1}, y\mathbf{p}_{2k}), \ldots, (y\mathbf{p}_{m-1}, y\mathbf{p}_m)$

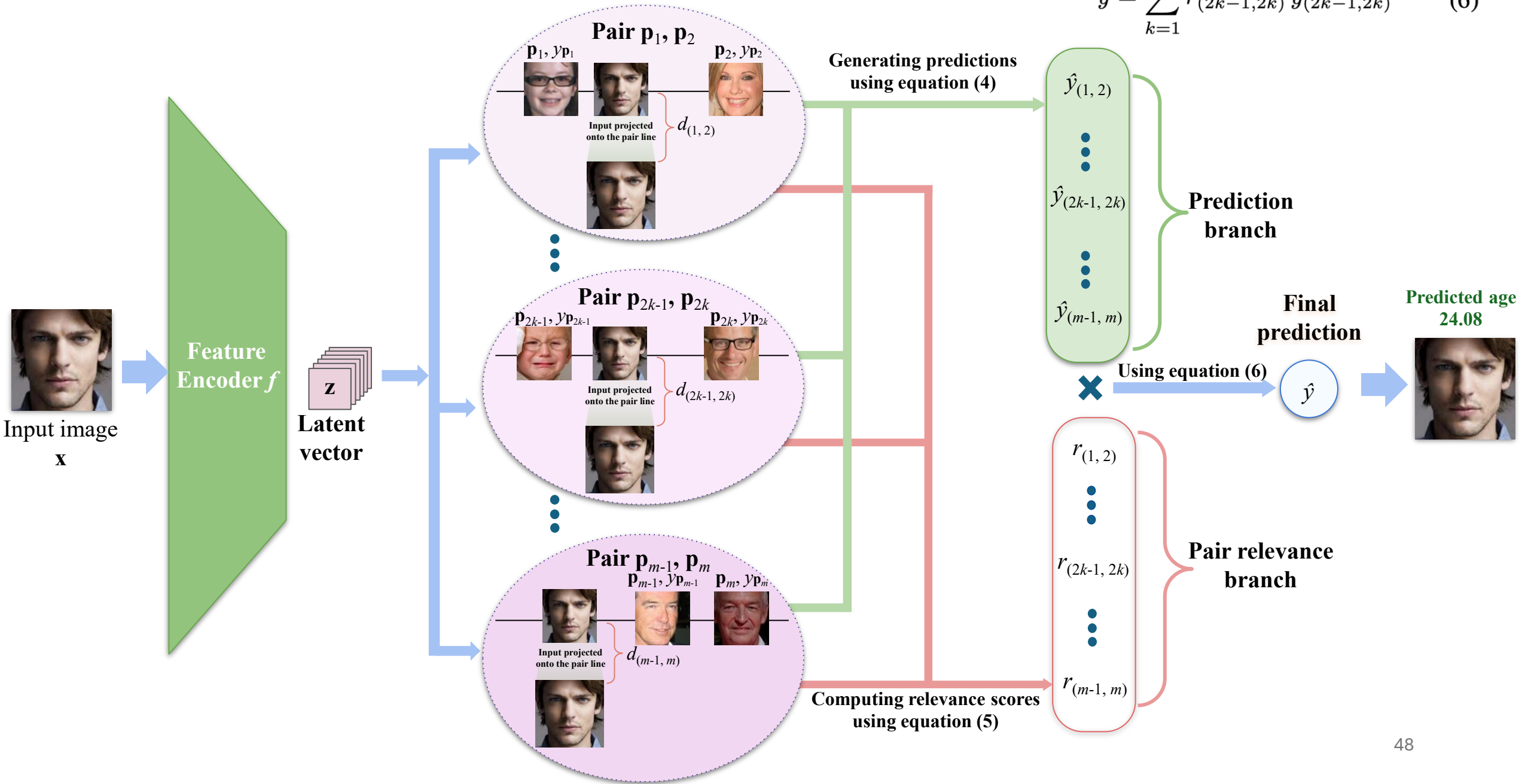
$$r_{(2k-1,2k)} = \frac{\exp\left[(-d_{(2k-1,2k)} + g_{(2k-1,2k)})/\tau\right]}{\sum_{k'=1}^{m/2} \exp\left[(-d_{(2k'-1,2k')} + g_{(2k'-1,2k')})/\tau\right]} \quad (5)$$

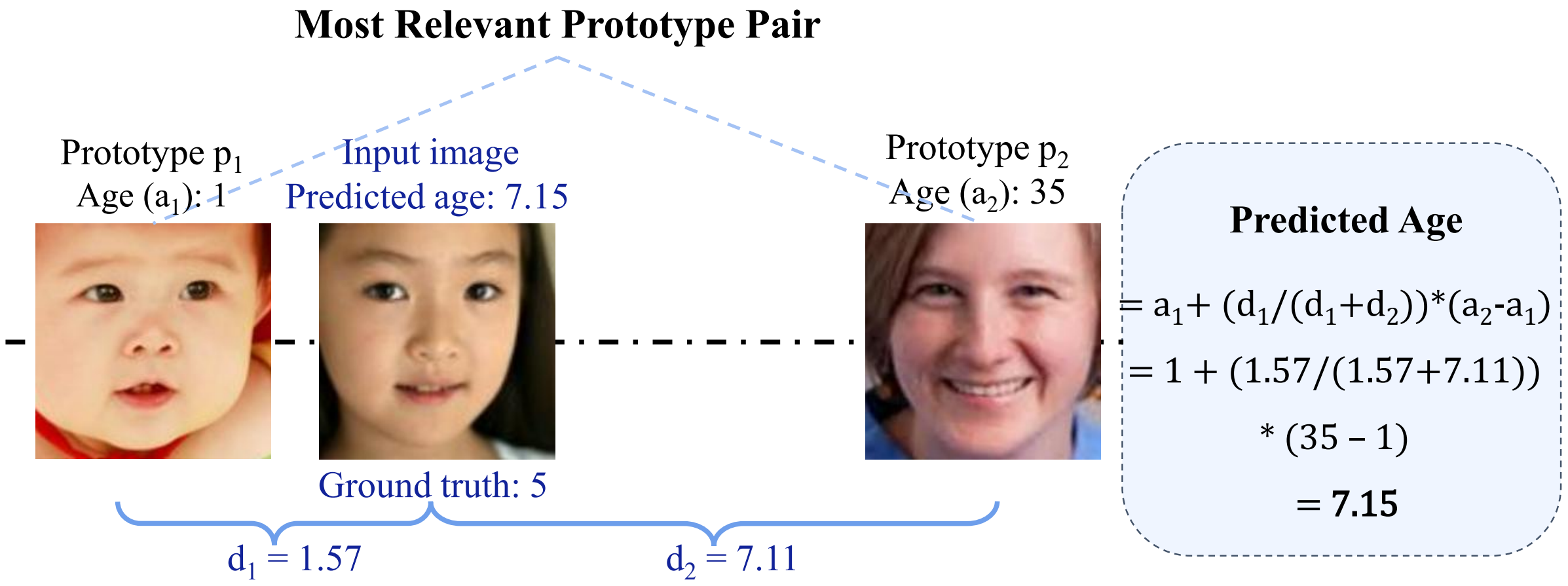


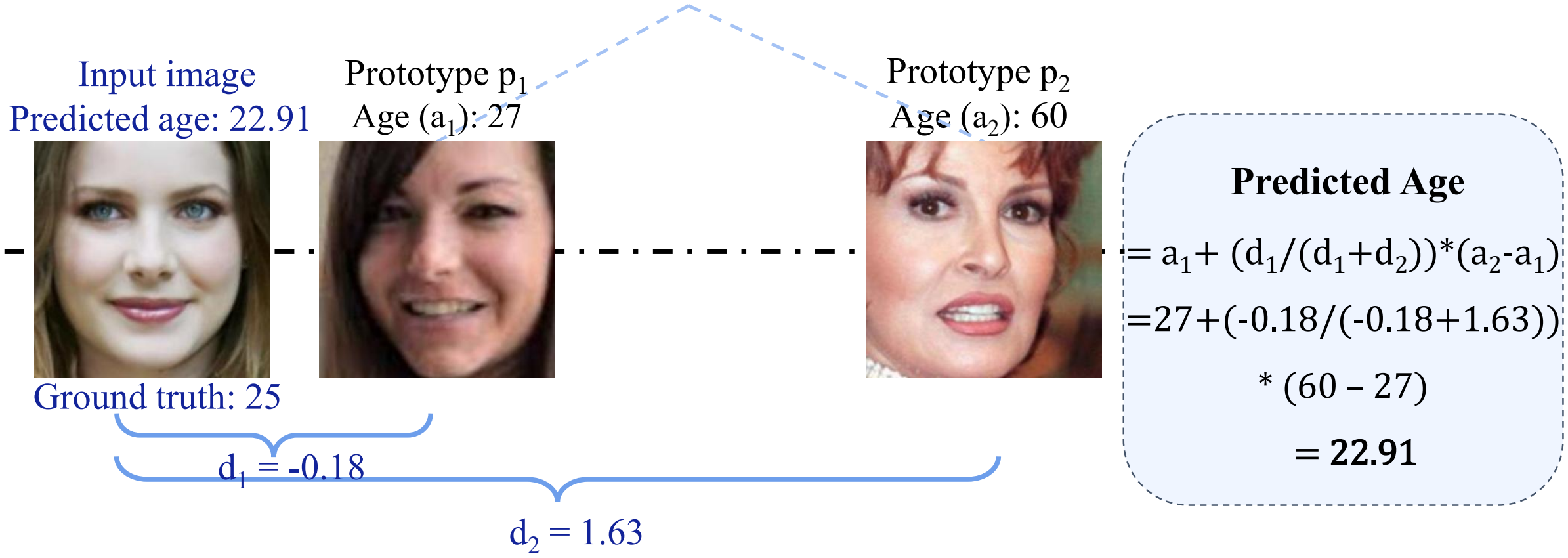
Prototypical pairs and corresponding labels:

$(\mathbf{p}_1, \mathbf{p}_2), \ldots, (\mathbf{p}_{2k-1}, \mathbf{p}_{2k}), \ldots, (\mathbf{p}_{m-1}, \mathbf{p}_m)$, and $(y_{\mathbf{p}_1}, y_{\mathbf{p}_2}), \ldots, (y_{\mathbf{p}_{2k-1}}, y_{\mathbf{p}_{2k}}), \ldots, (y_{\mathbf{p}_{m-1}}, y_{\mathbf{p}_m})$

$$\hat{y} = \sum_{k=1}^{m/2} r_{(2k-1,2k)}\, \hat{y}_{(2k-1,2k)} \qquad (6)$$

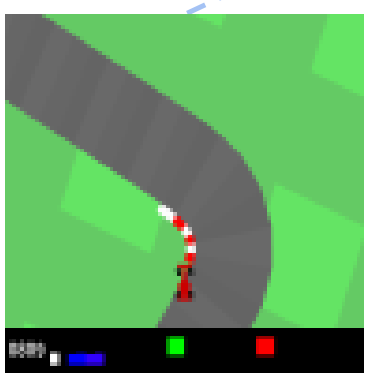



Most Relevant Prototype Pair
Prototype p_1
Age (a_1): 1
Input image
Predicted age: 7.15
Prototype p_2
Age (a_2): 35
Ground truth: 5
d_1 = 1.57
d_2 = 7.11
Predicted Age
$= a_1 + (d_1/(d_1+d_2))*(a_2-a_1)$
$= 1 + (1.57/(1.57+7.11))$
$* (35 - 1)$
$= \mathbf{7.15}$

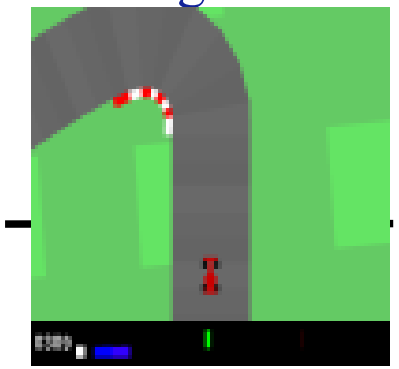

Most Relevant Prototype Pair
Input image
Predicted age: 22.91
Prototype p_1
Age (a_1): 27
Prototype p_2
Age (a_2): 60
Ground truth: 25
d_1 = -0.18
d_2 = 1.63
Predicted Age
$= a_1 + (d_1/(d_1+d_2))*(a_2-a_1)$
$=27+(-0.18/(-0.18+1.63))$
$* (60 - 27)$
$= 22.91$

Most Relevant Prototype Pair for steering

Prototype p_1
Steering (a_1): -0.93

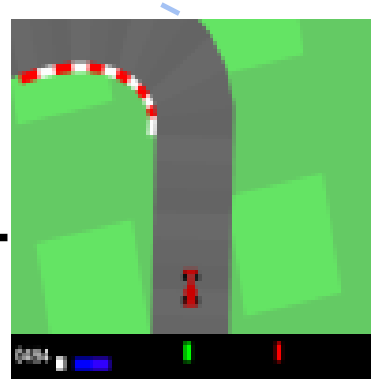


Input state
Predicted steering: -0.34



Original Steering: -0.39

Prototype p_2
Steering(a_2): -0.20



$d_1 = 90.35$

$d_2 = 21.65$

Predicted Steering

$= a_1 + (d_1/(d_1+d_2))*(a_2-a_1)$

$=-0.93+$

$(90.35/(90.35+21.65))$

$*(-0.20 - (-0.93))$

$=$ **-0.34**

Table 1: Age prediction results. Comparision of ProtoPairNet with baseline black-box models (without prototypes) in terms of MAE and R^2 scores across different architectures.

Architecture	ResNet50		EfficientNet-B0		VGG19		ViT-Small (Patch 16)	
	MAE	R^2 Score	MAE	R^2 Score	MAE	R^2 Score	MAE	R^2 Score
Baseline	4.71 ± 0.09	0.86 ± 0.004	4.82 ± 0.08	0.86 ± 0.004	4.73 ± 0.07	0.86 ± 0.005	4.81 ± 0.06	0.86 ± 0.004
ProtoPairNet (Ours)	$\mathbf{4.59 \pm 0.01}$	$\mathbf{0.87 \pm 0.001}$	4.88 ± 0.03	0.86 ± 0.0006	$\mathbf{4.57 \pm 0.01}$	$\mathbf{0.87 \pm 0.001}$	$\mathbf{4.63 \pm 0.04}$	$\mathbf{0.87 \pm 0.001}$

Table 2: Comparison of ProtoPairNet with other prototype-based ResNet50 models for regression using MAE and R^2 scores.

Model	MAE	R^2 score
HPN	5.08 ± 0.03	0.84 ± 0.001
INSightR-Net (1×1)	14.01 ± 0.01	-0.03 ± 0.001
INSightR-Net (full)	14.81 ± 0.40	-0.08 ± 0.07
ExPeRT	17.36 ± 2.79	-0.37 ± 0.41
ProtoPairNet (Ours)	$\mathbf{4.59 \pm 0.01}$	$\mathbf{0.87 \pm 0.001}$

Table 3: Car Racing results. Comparison of different architectures in terms of rewards.

Architecture	Reward
k-Means[21]	-2.09 ± 0.94
PW-Net*[21]	-9.48 ± 2.50
PW-Net[21]	220.61 ± 0.70
ProtoPairNet	$\mathbf{223.97 \pm 0.75}$
PPO black box agent	221.36 ± 0.96

Proto-RSet

- “Rashomon Sets for Prototypical-Part Networks: Editing Interpretable Models in Real-Time” (CVPR, 2025)

Rashomon Sets for Prototypical-Part Networks: Editing Interpretable Models in Real-Time

Jon Donnelly
Duke University
jon.donnelly@duke.edu

Zhicheng Guo
Duke University
zhicheng.guo@duke.edu

Alina Jade Barnett
Duke University
alina.barnett@duke.edu

Hayden McTavish
Duke University
hayden.mctavish@duke.edu

Chaofan Chen
University of Maine
chaofan.chen@maine.edu

Cynthia Rudin
Duke University
cynthia@cs.duke.edu

Abstract

Interpretability is critical for machine learning models in high-stakes settings because it allows users to verify the model's reasoning. In computer vision, prototypical part models (ProtoPNets) have become the dominant model type to meet this need. Users can easily identify flaws in ProtoPNets, but fixing problems in a ProtoPNet requires slow, difficult retraining that is not guaranteed to resolve the issue. This problem is called the "interaction bottleneck." We solve the interaction bottleneck for ProtoPNets by simultaneously finding many equally good ProtoPNets (i.e., a draw from a "Rashomon set"). We show that our framework – called Proto-RSet – quickly produces many accurate, diverse ProtoPNets, allowing users to correct problems in real time while maintaining performance guarantees with respect to the training set. We demonstrate the utility of this method in two settings: 1) removing synthetic bias in-

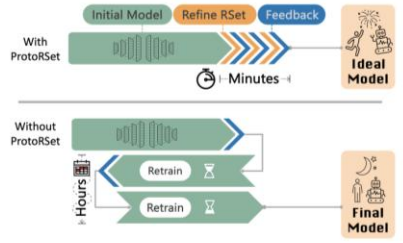
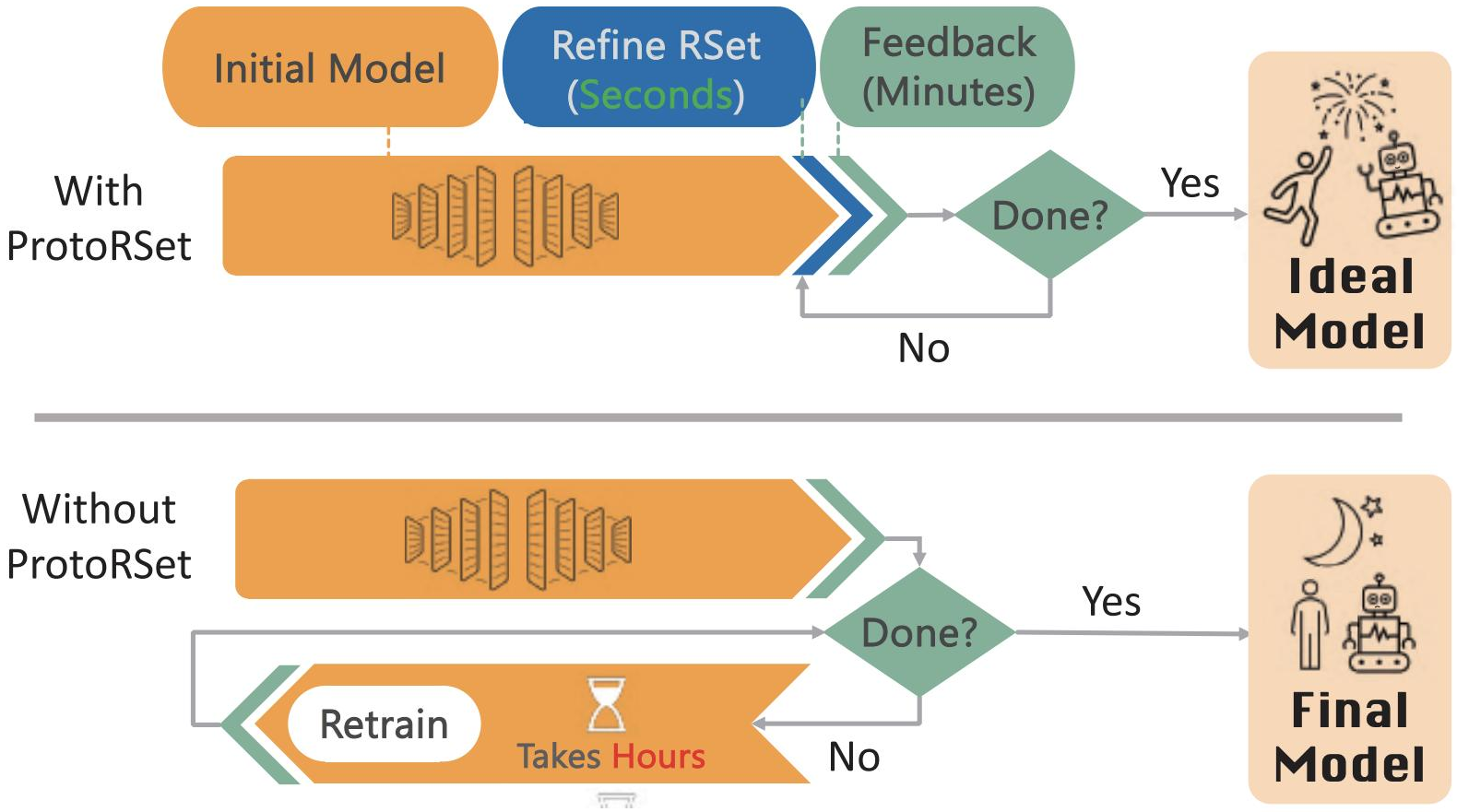


Figure 1. How Proto-RSet addresses the interaction bottleneck. (Bottom) Without Proto-RSet, incorporating user feedback such as "this prototype does not make sense, this would be a better option" requires practitioners to make complicated adjustments to their training regime and train a whole new model. This process can take days, and be prohibitively slow when multiple rounds

Proto-RSet


Initial Model
Refine RSet
(Seconds)
Feedback
(Minutes)
With
ProtoRSet
Done?
Yes
No
Ideal
Model
Without
ProtoRSet
Done?
Yes
Retrain
Takes Hours
No
Final
Model

Proto-RSet

- **Key observation**: a lot of feedback could be addressed by only adjusting the last layer
- **Excluding a prototype**: set its last layer connections to 0
- **Including a prototype**: upweight its last layer connection to its own class
- A naïve removal or upweighting of a prototype could destroy model performance!

Proto-RSet

- **Solution**: find a *Rashomon set* of prototype-based models that:
  - Share the same backbone and the same set of learned prototypes
  - But differ in their last layer weights!
- Rashomon set = set of near-optimal models on the training data

Proto-RSet

Approximate the training loss using its second-order Tayler expansion around $\mathbf{w}_h^*$, the optimal final layer:

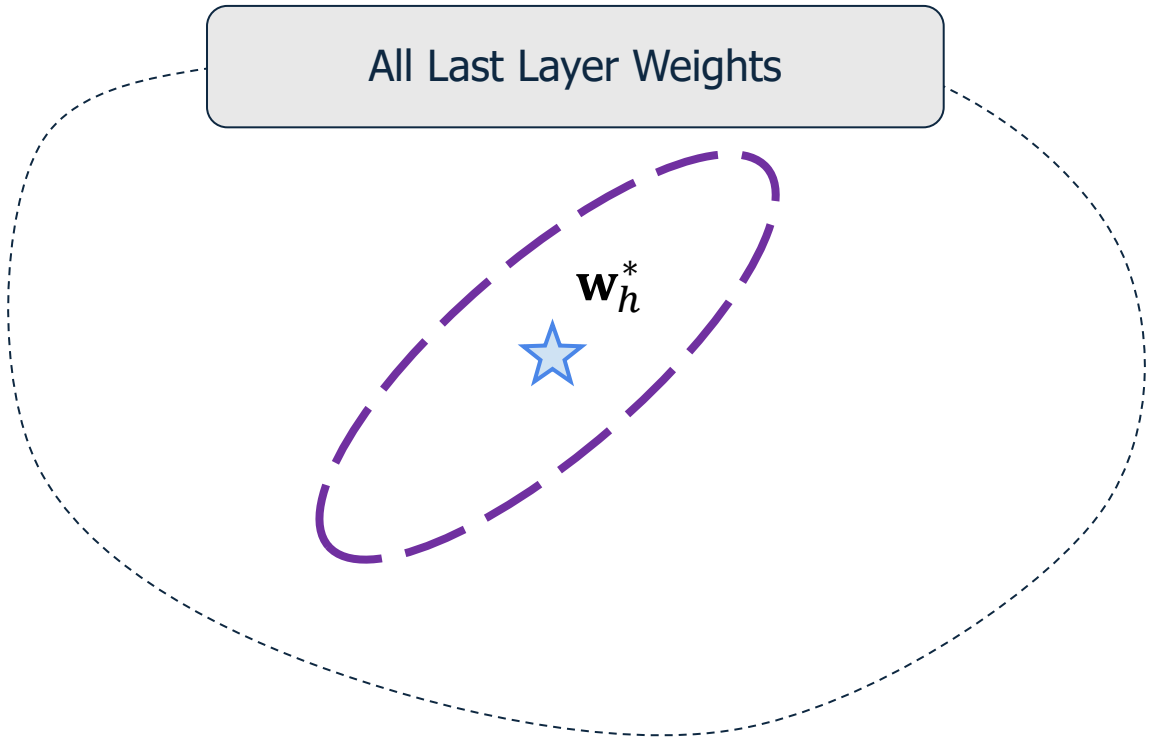
$$\begin{aligned}\bar{\ell}(\mathbf{w}_h) \approx & \bar{\ell}(\mathbf{w}_h^*) + (\nabla \bar{\ell}|_{\mathbf{w}_h^*})^T(\mathbf{w}_h - \mathbf{w}_h^*) \\ & + \frac{1}{2}(\mathbf{w}_h - \mathbf{w}_h^*)^T \mathbf{H}(\mathbf{w}_h - \mathbf{w}_h^*) \\ = & \bar{\ell}(\mathbf{w}_h^*) + \frac{1}{2}(\mathbf{w}_h - \mathbf{w}_h^*)^T \mathbf{H}(\mathbf{w}_h - \mathbf{w}_h^*)\end{aligned}$$

Rashmon set of near-optimal last layer weights is the set of last layer weights satisfying $\bar{\ell}(\mathbf{w}_h) \leq \theta$

This can be approximated using an ellipsoid:

$$\frac{1}{2(\theta - \bar{\ell}(\mathbf{w}_h^*))}(\mathbf{w}_h - \mathbf{w}_h^*)^T \mathbf{H}(\mathbf{w}_h - \mathbf{w}_h^*) \leq 1$$

Proto-RSet

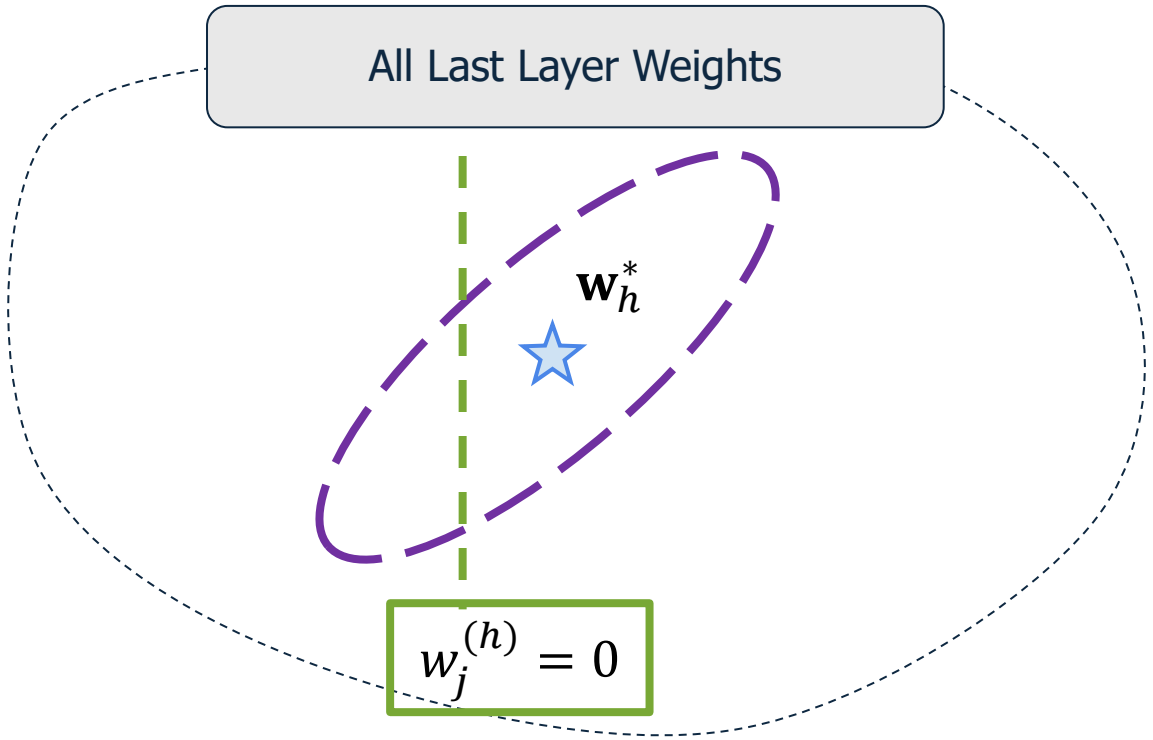


Rashmon set of near-optimal last layer weights is the set of last layer weights satisfying $\bar{\ell}(\mathbf{w}_h) \leq \theta$

This can be approximated using an ellipsoid:

$$\frac{1}{2(\theta - \bar{\ell}(\mathbf{w}_h^*))}(\mathbf{w}_h - \mathbf{w}_h^*)^T \mathbf{H}(\mathbf{w}_h - \mathbf{w}_h^*) \leq 1$$

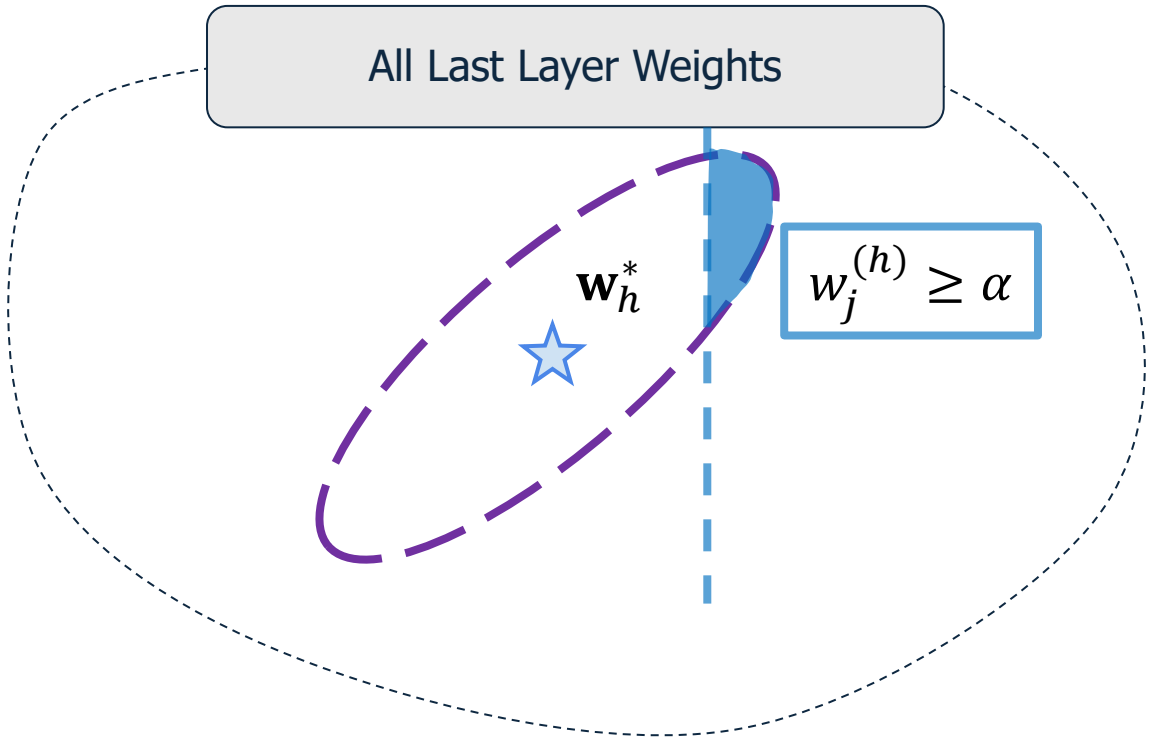
Proto-RSet



Remove the *j*-th prototype:

- Compute the intersection of the the ellipsoid with the hyperplane $w_j^{(h)} = 0$
- Still an ellipsoid!

Proto-RSet

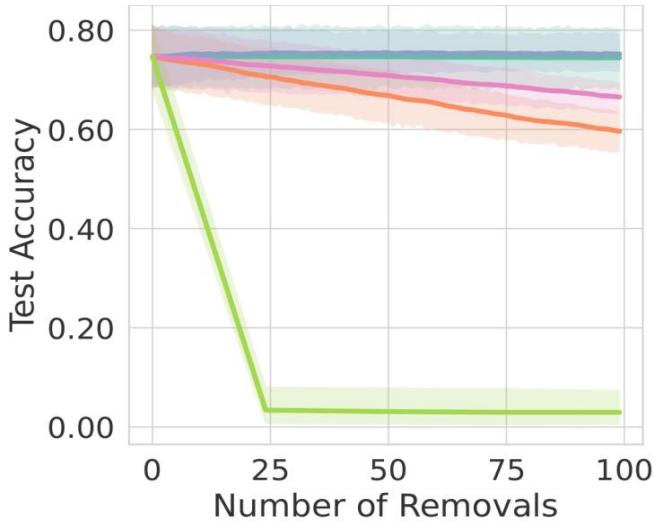


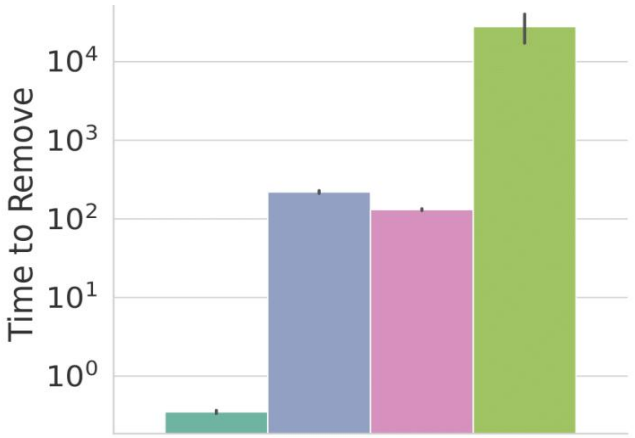
Upweight the *j*-th prototype:

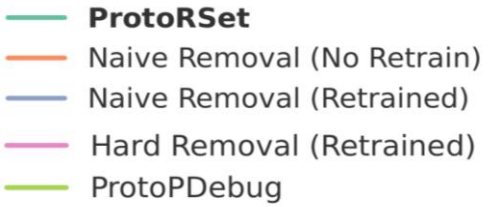
- Compute the intersection of the ellipsoid with a half space $w_j^{(h)} \geq \alpha$

Proto-RSet

- **User study**: remove prototypes that correspond to synthetic confounders added to a dataset

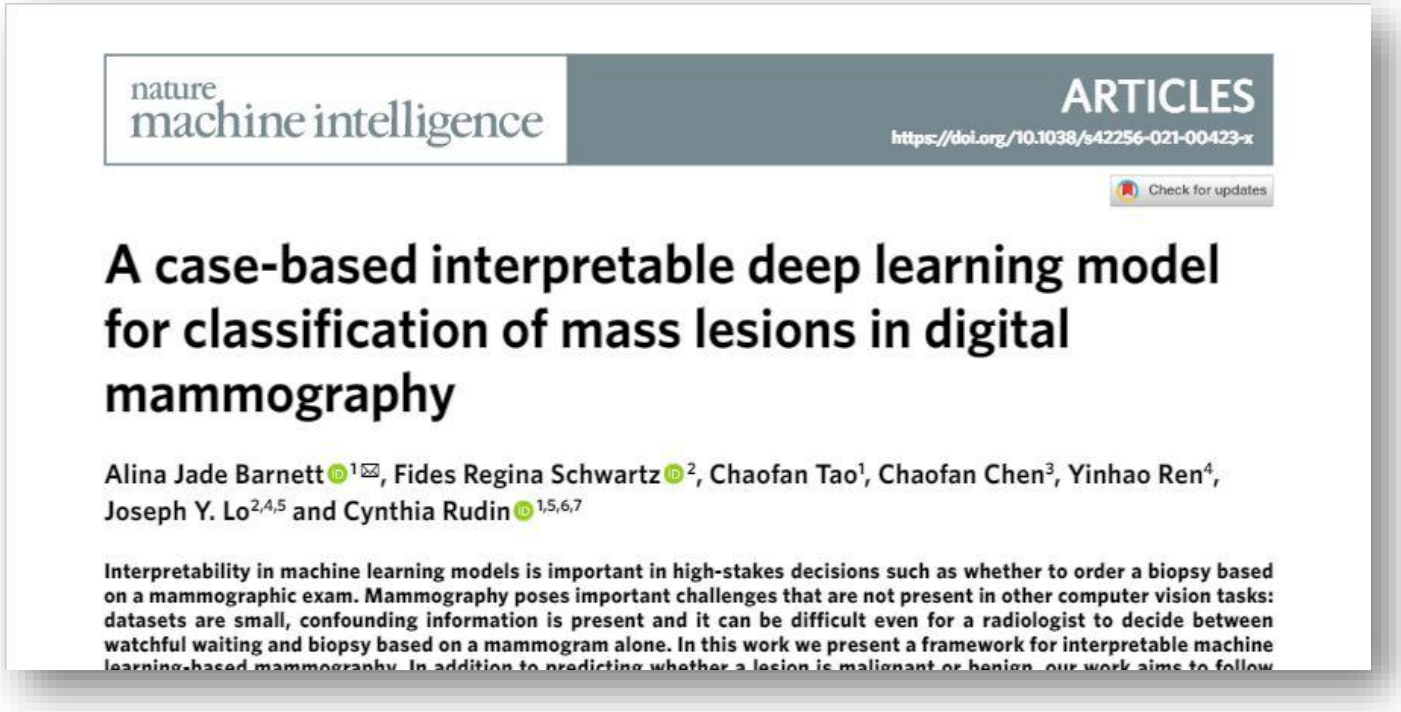






Application: mammography classification

- "A Case-based Interpretable Deep Learning Model for Classification of Mass Lesions in Digital Mammography" (*Nature Machine Intelligence*, 2021)



nature machine intelligence

ARTICLES

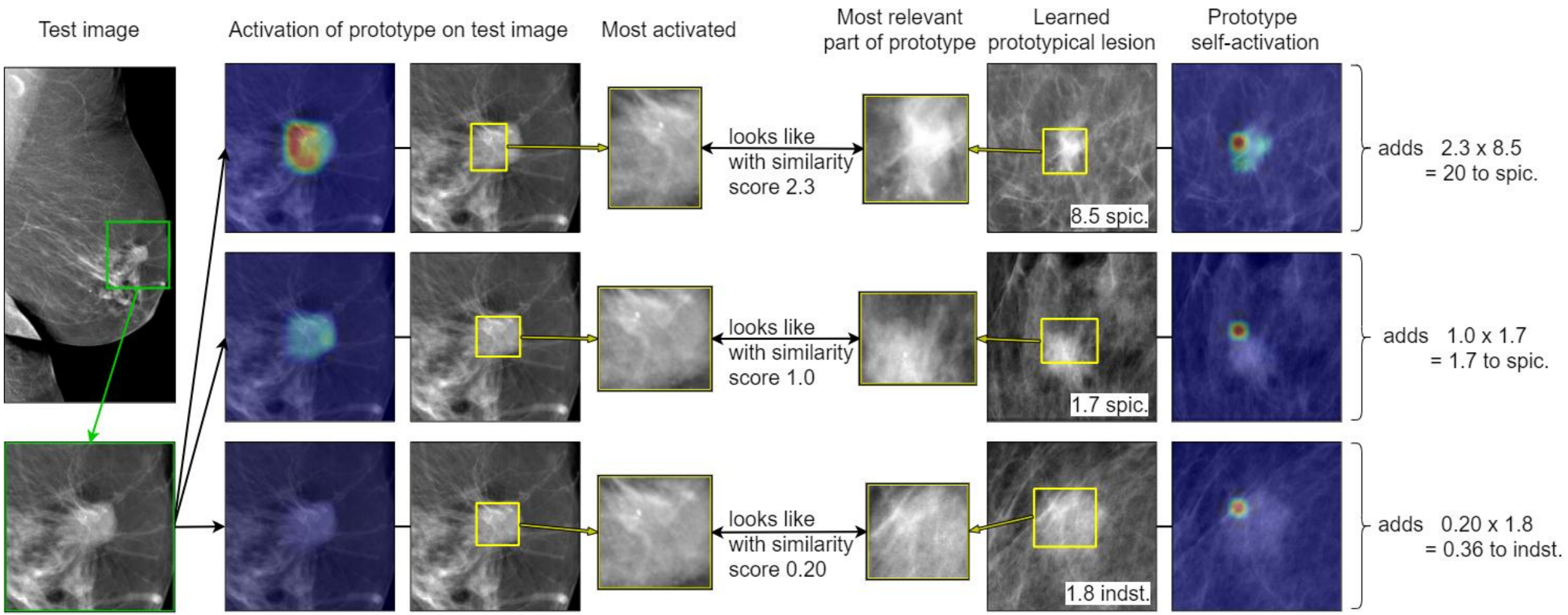
https://doi.org/10.1038/s42256-021-00423-x

Check for updates

A case-based interpretable deep learning model for classification of mass lesions in digital mammography

Alina Jade Barnett[1]✉, Fides Regina Schwartz[2], Chaofan Tao[1], Chaofan Chen[3], Yinhao Ren[4], Joseph Y. Lo[2,4,5] and Cynthia Rudin[1,5,6,7]

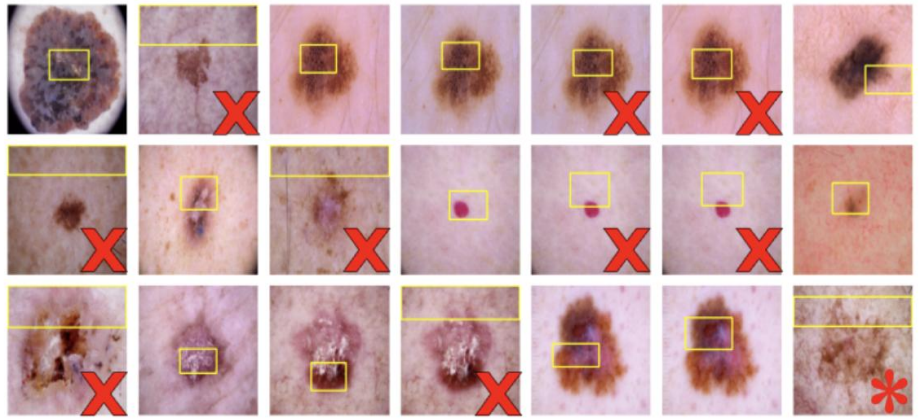
Interpretability in machine learning models is important in high-stakes decisions such as whether to order a biopsy based on a mammographic exam. Mammography poses important challenges that are not present in other computer vision tasks: datasets are small, confounding information is present and it can be difficult even for a radiologist to decide between watchful waiting and biopsy based on a mammogram alone. In this work we present a framework for interpretable machine learning-based mammography. In addition to predicting whether a lesion is malignant or benign, our work aims to follow


Test image
Activation of prototype on test image
Most activated
Most relevant part of prototype
Learned prototypical lesion
Prototype self-activation
looks like with similarity score 2.3
8.5 spic.
adds 2.3 x 8.5 = 20 to spic.
looks like with similarity score 1.0
1.7 spic.
adds 1.0 x 1.7 = 1.7 to spic.
looks like with similarity score 0.20
1.8 indst.
adds 0.20 x 1.8 = 0.36 to indst.

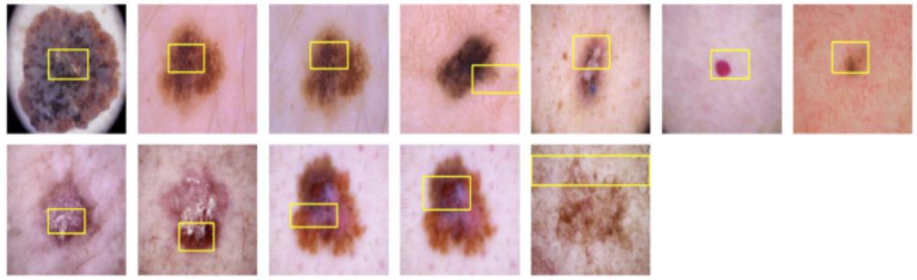
Application: skin cancer classification

- Applied Proto-RSet to a ProtoPNet trained on HAM10000 dataset for skin cancer classification to remove confounding prototypes (e.g., those that did not focus on the lesion)
- Removed 9 out of 21 prototypes
- Improved test accuracy!

Test accuracy: 70.4%



Test accuracy: 71.0%



Acknowledgments

- Funding support from National Science Foundation (NSF):
  - NSF 2442039
  - NSF 2218063
  - NSF 2244117
  - NSF 2416915
  - NSF 1849227