

Argumentative Interpretable Image Classification

Hamed Ayooobi¹, Nico Potyka² and Francesca Toni¹

¹Department of Computing, Imperial College London, United Kingdom

²Cardiff University, United Kingdom

Abstract

We propose *ProtoSpArX*, a novel interpretable deep neural architecture for image classification in the spirit of prototypical-part-learning as found, e.g. in ProtoPNet. While earlier approaches associate every class with multiple prototypical-parts, ProtoSpArX uses *super-prototypes* that combine prototypical-parts into single class representations. Furthermore, while earlier approaches use interpretable classification layers, e.g. logistic regression in ProtoPNet, ProtoSpArX improves accuracy with multi-layer perceptrons while relying upon an interpretable reading thereof based on a form of *argumentation*. ProtoSpArX is customisable to user cognitive requirements by a process of sparsification of the multi-layer perceptron/argumentation component. Also, as opposed to other prototypical-part-learning approaches, ProtoSpArX can recognise spatial relations between different prototypical-parts that are from various regions in images, similar to how CNNs capture relations between patterns recognized in earlier layers.

Keywords

Interpretable Image Classification, Argumentation, Prototypical-Parts Learning, XAI

1. Introduction

Deep neural architectures are successful in various tasks, but tend to be mostly inscrutable black-boxes. In high-stakes settings, interpretability is crucial and interpretable models are advocated over black-boxes, especially if they achieve comparable performance [2]. Prototypical-part learning for image classification amounts to learning prototypical-parts of classes in images by introducing a human-interpretable *prototype layer* between the *convolutional backbone* (intuitively, it learns patterns in the image space) and the *classification* component (intuitively, it uses the patterns identified by the backbone to classify an image) of convolutional neural networks [3]. Prototypical-parts can be seen as patches in images, like the

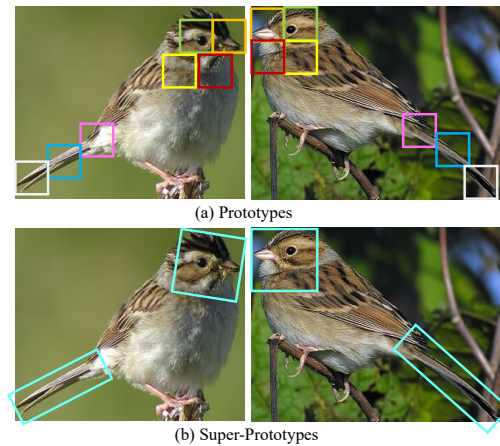


Figure 1: Conventional prototypes (a) versus the proposed super-prototypes (b) for a sample in the CUB dataset [1].

ArgXAI-24: 2nd International Workshop on Argumentation for eXplainable AI

✉ h.ayooobi@imperial.ac.uk (H. Ayooobi); PotykaN@cardiff.ac.uk (N. Potyka); f.toni@imperial.ac.uk (F. Toni)

🌐 <https://profiles.imperial.ac.uk/h.ayooobi> (H. Ayooobi); <https://profiles.cardiff.ac.uk/staff/potykan> (N. Potyka);

<https://www.doc.ic.ac.uk/~ft/> (F. Toni)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

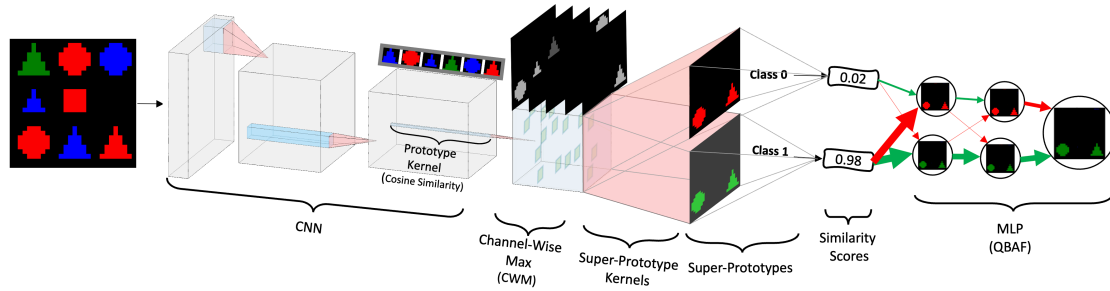


Figure 2: Architecture of ProtoSpArX (see Section 4 for the details), illustrated with a sample from the SHAPES dataset [6].

beak or tail of a bird (see Figure 1 (a)). The prototype layer determines the similarity between prototypical-parts and patches in the latent space that the convolutional backbone maps to. Even though some prototypical-parts may correspond to background patches that are meaningless for humans (rather than exclusively meaningful parts in images as in Figure 1 (a)), they allow making transparent classifications, based on clearly defined prototypes, if the classification component is interpretable.

We propose *ProtoSpArX* (Section 4, overviewed in Figure 2), a novel interpretable deep neural architecture for image classification in the spirit of prototypical-part-learning. Similar to ProtoPShare [4] and ProtoTrees [5], ProtoSpArX shares prototypes among classes. However, while these and other prototypical-part-learning approaches associate every class with multiple prototypical parts, ProtoSpArX summarizes them in a single *super-prototype* per class that encodes spatial relations among them (see Figure 1 (b) for an illustration).

The use of super-prototypes allows capturing spatial relations between prototypical parts similar to how CNNs capture relations between patterns recognized in earlier layers. As we will show in the experiments with the SHAPES dataset [6], these relations are essential for some classification tasks but state-of-the-art prototypical-part-learning approaches are unable to capture them. For example, in Figure 2, a positive example (Class 1) has a triangle in the left column and a circle in the right column on the same row. Merely recognizing prototypical-parts for triangles and circles in the input image (as in other prototypical-part-learning approaches) is insufficient for determining the class label in this example. ProtoSpArX effectively tackles this challenge by encoding the spatial relations between distinct prototypical-parts using the super-prototype kernels.

The classifier component in ProtoSpArX is a quantitative bipolar argumentation framework (QBAF) that is trained using the SpArX methodology of [7]. Intuitively, the QBAF uses weighted attacks and supports between super-prototypes and meta-arguments (latent arguments attacked and supported by super-prototypes or other meta-arguments) to classify an image. This is indicated by the red and green arrows in Figure 2¹.

We show experimentally (Sections 5 and 6) that ProtoSpArX outperforms the state-of-the-art prototypical-part-learning models ProtoPNet [3], ProtoTree [5], ProtoPShare [4], ProtoPool [8] and PIP-Net [9] in terms of classification accuracy and the ability to encode and detect spatial relations in images, supported by a number of ablations and the study of the cognitive complexity

¹Please note that the colour of the shapes in the input image has no relation with or bearing on the colours of the super-prototypes and edges in the QBAF, indicating attack and support (see Section 4 for details).

of local explanations derived from the sparsification of QBAFs obtained with ProtoSpArX.

2. Related Work

The problem of explaining image classifiers is well studied in the literature. Examples include feature attributions [10], attention maps [11] and counterfactual explanations [12]. While the former can be seen as *post-hoc explanations* that aim at explaining the decisions of a black-box classifier, there is also an increasing literature on interpretable-by-design approaches. One interesting interpretable direction is based on prototypical-part-learning [13]. These approaches were motivated by the observation that class-prototypes [14] for datasets with simple backgrounds (as in MNIST [15]) do not generalize well to natural images with more complex backgrounds. To overcome this problem, *ProtoPNet* [3] introduced prototypical parts for capturing parts of the class (like the beak or tail of a bird) rather than the whole object (the bird). The original idea has been extended in various directions including prototypes that can be shared among classes [4], the integration of prototypical parts into decision trees [5] and improved similarity functions [8]. Our ProtoSpArX adds super-prototypes and uses bipolar quantitative argumentation to achieve a better tradeoff between classification performance and interpretability. Specifically, ProtoSpArX extends the *SpArX* approach [7], originally defined for MLPs with tabular data only, to the setting of prototypical-part-learning with images.

Several other argumentation-based forms of explainability have been proposed, we refer to [16] for an overview. Other works combine argumentation and image classification, e.g. [17, 18] for explaining the outputs of CNNs and [19] to obtain an interpretable image classifier. ProtoSpArX may also be deemed neuro-symbolic as it combines, end-to-end (see Figure 2), neural components (the convolutional backbone, the prototype kernels, and the super-prototype kernels) with symbolic argumentation frameworks (QBAFs) drawn from MLPs. However, whereas recent neuro-symbolic systems often combine purely symbolic with purely neural systems [20, 21], ProtoSpArX is based on the observation that MLPs can be seen as QBAFs and vice versa [22, 7]. We keep the reasoning process in QBAFs interpretable by sparsification, as in [7].

3. Preliminaries

We build up on SpArX [7], a post-hoc explanation method that aims at generating structurally faithful explanations for MLPs. SpArX exploits that MLPs can be understood as Quantitative Bipolar Argumentation Frameworks (QBAFs) [22]. QBAFs can be seen as graphical reasoning models whose nodes represent *arguments* and whose edges represent *attack* or *support* relations between the arguments, each with a (negative or positive, respectively) intensity value [23, 24, 25, 26].

Arguments in QBAFs are abstract entities (what makes them arguments is that they are in dialectical relationships). To capture MLPs as in [22], these abstract arguments represent input features, hidden neurons and output classifications, and the graphical structure of QBAFs mirror the MLP.

This correspondence allows representing MLPs faithfully by QBAFs, but the QBAF representation is not useful for interpretability and explainability, because the QBAF has the same size as the original MLP. Thus, SpArX clusters neurons with similar activations and summarizes each cluster as a single argument [7]. Experiments with tabular data show that SpArX can give explanations that are both sparse and faithful [7].

In this work, we extend SpArX to make ProtoSpArX interpretable and explainable. An illustration is given in Figure 2: neurons in the MLP component of ProtoSpArX are treated as arguments, alongside the similarity scores from the super-prototypes, which serve as the input features for the MLP in our architecture (see the examples in Section 4 for further details on this illustration). Similarly to the original SpArX, we experiment with sparsification by various compression ratios (Section 6.4), showing that ProtoSpArX can provide explanations that are both sparse and faithful for image classification.

4. Method

Figure 2 shows the architecture of ProtoSpArX. ProtoSpArX consists of a convolutional backbone f with weights W^{conv} , a prototype layer $\mathcal{P}$, a Channel-Wise Max (CWM) layer $\mathcal{CWM}$, a Super-Prototype kernel $\mathcal{SP}$ followed by an MLP $\mathcal{MLP}$ with weights $W^{\mathcal{MLP}}$, mapped onto a QBAF for interpretability and explainability purposes. We discuss each component in turn, assuming that inputs are images and the classification task amounts to predicting a class in the set K ($|K| \geq 2$).

4.1. Prototypes

Let $z = f(x)$ be the convolutional output for an input image x , where the output tensor z has shape $H \times W \times D$ with height H , width W and D channels. This output tensor serves as input to the prototype layer, $\mathcal{P}$, which represents prototypical-parts. $\mathcal{P}$ consists of N prototypes $P = \{p_i\}_{i=1}^N$ with shapes $H_1 \times W_1 \times D$ (we have used $H_1 = W_1 = 1$ in all experiments). For each prototype $p_i \in P$ and every $H_1 \times W_1 \times D$ sub-tensor z_j of z , the prototype layer $\mathcal{P}$ computes the cosine similarity

$$\mathcal{CS}(p_i, z_j) = \frac{p_i \cdot z_j}{\|p_i\| \|z_j\|} \quad (1)$$

and outputs a similarity map

$$\mathcal{SM}_i = \mathcal{CS}_{z_j \in z}(p_i, z_j) \quad (2)$$

with shape $H \times W$ for each prototype $p_i \in P$. Intuitively, $\mathcal{SM}_i$ indicates how similar the prototypical-part p_i is to patches of the input image x in the latent space. We implemented $\mathcal{SM}$ using the 2D convolution operator $*$. It generates $\mathcal{SM}_i$ by convoluting the normalized convolutional output $\hat{z} = \frac{z}{\|z\|} = \left[\frac{z_j}{\|z_j\|} \right]_{z_j \in z}$ with a normalized prototype kernel $\hat{p}_i = \frac{p_i}{\|p_i\|}$, $\mathcal{SM}_i = \hat{z} * \hat{p}_i$. Since cosine similarity is used for the prototype layer, the values in similarity maps can be both positive and negative in the range $[-1, 1]$. The output dimensions of the prototype layer are $H \times W \times N$.

4.2. Channel-Wise Max

The Channel-Wise Max layer aims to both localize and extract the max value of each similarity map while maintaining its dimensions. $\mathcal{CWM}$ takes the similarity maps as input and extracts the maximum value from each input channel by passing the maximum value and setting all other values to zero while preserving the input dimensions. Formally, for every similarity value $s \in \mathcal{SM}_i$, the i^{th} similarity map, the channel-wise max filter $\mathcal{CWM}_i$ retains the highest value $s_{max} = \max(\mathcal{SM}_i)$ within the map and assigns a value of zero to the remaining elements:

$$\mathcal{CWM}_i = \begin{cases} s_{max} & \text{if } s = \max(\mathcal{SM}_i); \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

The output dimensions of $\mathcal{CWM}$ are still $H \times W \times N$.

4.3. Super-Prototypes and Similarity Scores

The super-prototypes kernel takes the output of the channel-wise max layer as input and provides a single representation per class. This is done in three steps.

In the first step, for each class $k \in K$, M linear combinations of the channel-wise max filters, denoted by $\mathcal{LC}_i^k$ where $i \in \{1, \dots, M\}$, are learned. Here, M is a customisable hyper-parameter of the model ($M = 32$ achieved the best results in the experiments). Formally:

$$\mathcal{LC}_i^k = \sum_{j=1}^N w_j^{\mathcal{LC}_i^k} \cdot \mathcal{CWM}_j \quad (4)$$

where $w_j^{\mathcal{LC}_i^k}$ is a trainable scalar weight. We let $W^{\mathcal{LC}}$ denote the vector summarizing all these weights. This operation can be implemented with M convolutions with kernel shape $1 \times 1 \times N$ using the N channel-wise max filters as input.

In the second step, the super-prototypes are constructed. Each linear combination $\mathcal{LC}_i^k$ is then multiplied by a trainable weight matrix $W_i^{\mathcal{SP}^k}$ with shape $H \times W$ to obtain a single super-prototype for each class from the M linear combinations. This means that the number of super-prototypes is equal to the number of classes $|K|$. Each super-prototype $\mathcal{SP}^k$ is then computed as follows:

$$\mathcal{SP}^k = \sum_{i=1}^M \mathcal{LC}_i^k \odot W_i^{\mathcal{SP}^k}, \quad (5)$$

where $\odot$ denotes element-wise product. Each super-prototype has the shape $H \times W$. By utilizing the receptive field of the convolutional output f to rescale the similarity maps $\mathcal{SM}$ to the input dimensions, the super-prototypes can be visualized on the input image x employing Equation 5, as illustrated next.

Example 1. Figure 2 illustrates the visualization of the super-prototypes on the input image, where the colours indicate support (green) for Class 1 at the bottom and attack (red) against Class 0 at the top. Note that, since we are dealing with binary classification, the supporting regions for accepting one class are the attacking regions for accepting the other class. Also, the

colours in the input images are irrelevant to the classification task which associates Class 1 to images with a triangle in the left column and a circle in the right column on the same row, no matter their colour.

In the third and final step, a single similarity score ss_k is computed for each super-prototype by summing up the values $sp \in \mathcal{SP}^k$:

$$ss^k = \sum_{sp \in \mathcal{SP}^k} sp. \quad (6)$$

Equations 5 and 6 can be simultaneously implemented by employing $|K|$ convolutions with a kernel shape of $H \times W \times M$, while taking the M linear combinations for each class as input.

4.4. Classifier Layer

Using the similarity scores as input, $\mathcal{MLP}$ is used for classification. After the training phase, $\mathcal{MLP}$ is converted to a QBAF (c.f., Section 3 – this involves sparsifying the underlying $\mathcal{MLP}$ and then translating it to a QBAF). The obtained QBAF can provide reasons for and against assigning an input x to a specific class, making ProtoSpArX interpretable as illustrated next.

Example 2. The (sparsified) 1-hidden layer-MLP/QBAF in Figure 2 can be interpreted as follows:

- Super-prototype of Class 1 supports and attacks, with high intensity, the arguments corresponding to, respectively, the bottom and top neurons in the hidden layer;
- Conversely, the super-prototype for Class 0 attacks and supports, with low intensity, the same arguments;
- The hidden clusters and output neurons are visualized using the super-prototypes they “propagate” through the MLP, in the sense that these super-prototypes support them, e.g. the super-prototype for Class 0 supports the top cluster in the hidden layer and the predicted Class 1 is supported by the super-prototype for Class 1.

Overall, this interpretation indicates that the predicted Class 1 for the input image is supported by the presence of a circle in the bottom left corner and a triangle in the bottom right corner, while also pointing to the reasoning of the MLP in terms of the super-prototypes used.

5. Training ProtoSpArX

Unlike other prototypical-part-learning approaches, the training phase of ProtoSpArX is done in one step. This means that all amongst the prototype layer, the super-prototype kernels and the classifier are trained at once without a need for freezing the weight of the classifier first and fine-tuning it later. For the i^{th} data point in a dataset of size n , with the data point belonging to class label $y_i \in K$ (where K is the set of class labels), the target class super-prototype should obtain a high similarity score ss_{y_i} . Moreover, the corresponding similarity scores for the super-prototypes of other classes ($\{ss_k\}_{k=1, k \neq y_i}^{|K|}$) should be low. Simultaneously, the output of the classifier should be 1 for the target class y_i and 0 for the other classes. Therefore, we integrate in the loss function two components $L_{\mathcal{SP}}$ and L_{cls} for the corresponding objectives.

Definition 1. The *total loss function* $\mathcal{L}$ that we aim to minimize is:

$$\mathcal{L} = L_{CE} + L_{SP} \quad (7)$$

where L_{CE} is the Cross-Entropy loss and L_{SP} is a regularization term that aims at associating super-prototypes with their associated classes by penalizing the similarity to wrong classes and rewarding the similarity to the correct class :

$$L_{CE} = \sum_{i=1}^n \text{CrsEnt}(G(x_i), y_i), \quad (8)$$

$$L_{SP} = \sum_{i=1}^n \left(\left(\sum_{\substack{k=1 \\ k \neq y_i}}^{|K|} ss_k \right) - ss_{y_i} \right); \quad (9)$$

where $G(x_i)$ denotes the output of ProtoSpArX.

Given the definition of total loss function $\mathcal{L}$, we then use the Adam optimizer [27] to tune the convolutional weights W^{conv} , prototypes $\mathcal{P}$, linear combination weights $W^{\mathcal{LC}}$, super-prototype weights W^{SP} , and MLP weights $W^{\mathcal{MLP}}$ in an end-to-end fashion to minimize $\mathcal{L}$:

$$\min_{W^{conv}, \mathcal{P}, W^{\mathcal{LC}}, W^{SP}} \mathcal{L}(W^{conv}, \mathcal{P}, W^{\mathcal{LC}}, W^{SP}) \quad (10)$$

Finally, for the projection of prototypes, we follow the same approach as ProtoPNet [3] to push the prototypes to the latent representation of the closest image patch from the input space in the convolutional output so that each prototype has a global interpretable representation.

6. Experiments

We have compared our approach with the state-of-the-art prototypical-part-learning models ProtoPNet [3], ProtoTrees [5], ProtoPShare [4], ProtoPool [8] and PIP-Net [9]. We have conducted four sets of experiments to evaluate the classification performance (Section 6.1), the role of each layer on the model’s performance by an ablation study (Section 6.2), the ability to encode and detect spatial relationships in the input (Section 6.3), and the cognitive complexity of explanations naturally drawn from ProtoSpArX (Section 6.4). Notice that, for all the experiments, we use classification accuracy as our performance measure, as is the case with the baselines.

For all the experiments, we have used CUB-200-2011 (CUB)[1] and Stanford Cars (Cars) [28], which are the standard benchmarks for prototypical-part learning models. To assess the ability to encode spatial relationships, we use the SHAPES dataset [6] adapted to binary classification.

Method	Accuracy		
	CUB	Cars	SHAPES
ProtoPNet	79.2 ± 0.1	86.1 ± 0.1	51.1 ± 0.7
ProtoPShare	74.7 ± 0.2	86.4 ± 0.2	50.4 ± 0.8
ProtoPool	80.3 ± 0.2	88.9 ± 0.1	50.8 ± 0.6
ProtoTrees	82.2 ± 0.7	86.6 ± 0.2	51.4 ± 0.7
PIP-Net	82.0 ± 0.3	86.5 ± 0.3	50.6 ± 0.6
ProtoSpArX	83.4 ± 0.2	89.3 ± 0.2	98.4 ± 0.2

Table 1: Accuracy of ProtoSpArX and other prototypical-part-learning methods for different datasets. (Best accuracy in **bold**)

6.1. Classification Performance

The first two columns in Table 1 show the accuracy of our method compared to the baselines, for CUB and Cars. For both datasets, our ProtoSpArX outperforms the other approaches.

6.2. Ablation study

Ablation studies on CUB and Cars in Table 2 show that ProtoSpArX achieves the best accuracy when employing super-prototypes atop the cosine similarity prototype layer, together with an MLP as classifier component. Alternatively, the L2-distance-based prototype layer, as utilized in ProtoPNet, can be employed in conjunction with a fixed logistic regression layer for classification (fine-tuned in the second training phase in ProtoPNet). Notably, ProtoSpArX surpasses the performance of state-of-the-art methods even when utilizing a fixed logistic regression layer, instead of an MLP as the classifier (but performs best with the MLP).

Super-Prototype	Prototype Layer	Classifier	Accuracy	
			CUB	Cars
—	L2	Fixed	79.2	86.1
—	L2	MLP	81.2	86.7
—	Cosine	Fixed	81.5	87.2
—	Cosine	MLP	81.8	87.8
✓	L2	Fixed	81.0	87.3
✓	L2	MLP	81.6	87.9
✓	Cosine	Fixed	82.7	88.9
✓	Cosine	MLP	83.4	89.3

Table 2: Ablation study with different prototype layers and classifiers with respect to a super-prototype kernel. (Best accuracy in **bold**)

6.3. Spatial Correlations

To assess whether different image classification methods can account for spatial relationships between prototypical-parts in images, we adapted the SHAPES dataset [6] as a benchmark. We randomly generated synthetic images containing 3×3 grids of circles, triangles, and squares in different colours (red, green, and blue), so that an image is assigned Class 1 if a triangle is located in the first column and a circle is located in the third column of the same row², and Class 0 otherwise. The resulting dataset comprises 10,000 28×28 images with balanced binary class labels. Figure 4 shows examples of images in the dataset. The first row contains images from class 1, where a triangle is located in the first column and a circle is located in the third column of the same row. The second row contains images from class 0, where this condition is not met.

The last column in Table 1 compares the accuracy of the baselines for this SHAPES dataset. ProtoSpArX, with an accuracy of $98.4\% \pm 0.2\%$, significantly outperforms all other approaches. The accuracy of the other approaches is around 50%, suggesting that these models are unable to infer class labels solely based on the presence of prototypes in images, being unable to infer information about the relative placement of the prototypical-parts in the images. ProtoSpArX addresses this limitation by using channel-wise max and super-prototypes, which enable the

²This criterion can be customized to reflect the user’s preferences. For example, the dataset could assign Class 1 to images with a square in the first column, a blue triangle in the second, and a red square in the third.

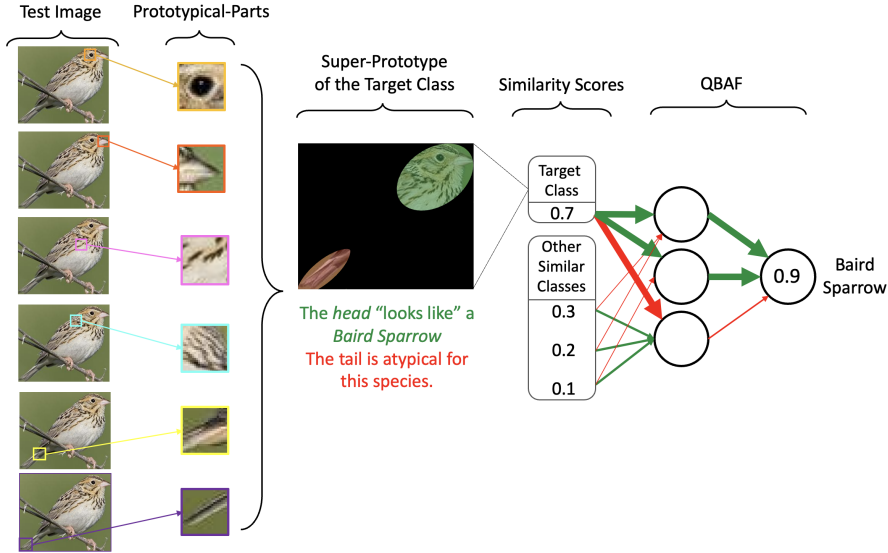


Figure 3: Example of ProtoSpArX explanation for a *Baird Sparrow* image from CUB dataset. The prototypical-parts are learned from the training image patches. The super-prototype highlights the supported regions with a green overlay and the attacked regions with a red overlay. The QBAF outlines the reasoning of the MLP while assigning the probability of 0.9 for classifying the input as *Baird Sparrow*.

model to infer the spatial correlation of different prototypical-parts in the image when needed for classification.

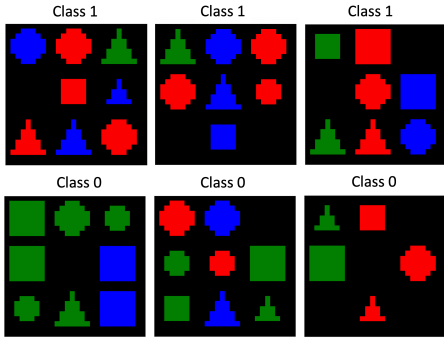


Figure 4: Examples from adapted SHAPES dataset, with binary class labels.

6.4. Cognitive Complexity

The combination of super-prototypes and QBAFs can serve as the basis for human-readable local explanations for the outputs of ProtoSpArX. Figure 2 showed a generated local explanation for a data point in SHAPES (see the examples in Section 4 for details on this illustration). Figure 3 illustrates a local explanation generated for a data instance from the CUB dataset, specifically for the target class “Baird Sparrow.” The green overlay on the super-prototype highlights the region in the input

	Method	# (Super-)Prototypes	
		CUB	Cars
Before Pruning	ProtoPNet	2000	2000
	ProtoPShare	2000	2000
	ProtoPool	202 / 202×5	195 / 195×5
	ProtoTrees	512	512
	PIP-Net	2000	2000
	ProtoArgNet	200	196
After Prune	ProtoPNet	2000	1960
	ProtoPShare	400	480
	ProtoPool	—	—
	ProtoTrees	202 / 202×3	195 / 195×3
	PIP-Net	495	515
	ProtoArgNet	—	—

Table 3: Comparison of the number of (super-)prototypes for different approaches. For ProtoPool and ProtoTrees, we also report the ensemble cases.

image that supports the correct classification, while the red region identifies the attacked or unsupported portion of the input. This super-prototype can be interpreted as the bird’s head resembles a “Baird Sparrow,” but its tail is atypical for this species. We have added this reading manually here for illustration, simulating how to read the super-prototypes. We leave the automatic generation of natural language interpretations of the super-prototypes and the QBAF for future work.

We can use the number of representative (super-)prototypes as a measure of the cognitive complexity of the explanations drawn from prototypical-part-learning methods. Table 3 compares the number of (super-)prototypes for each approach, before and after the pruning phase if applicable. Since ProtoPool and ProtoTrees use an ensemble of multiple models, we have also reported these cases. Like ProtoPool, our ProtoSpArX does not have an additional phase for pruning unnecessary prototypes. The number of super-prototypes in our approach is equal to the number of classes since ProtoSpArX has one super-prototype per class. Notice that using a fixed classification layer, as in ProtoPNet, for ProtoSpArX, the local explanations require only one super-prototype while other approaches need multiple prototypes.

The global cognitive complexity of ProtoSpArX should additionally include the number of hidden nodes in the MLP since each node in the resulting QBAF would be part of the explanation. This complexity can be controlled by sparsification as in SpArX [7], with a trade-off between compression ratio of the MLP classifier and accuracy of the resulting ProtoSpArX model. For illustration, considering a one-layer MLP and 10 arguments in the QBAF after the sparsification of the MLP, the cognitive complexity of the QBAF would be 210 and 206 for CUB and Cars, respectively.

7. Conclusion

We proposed ProtoSpArX, a novel prototypical-part-learning approach. ProtoSpArX learns a single super-prototype per class. The super-prototypes integrate multiple prototypical-parts shared between different classes into a representative prototype per class. It can be trained end-to-end and does not require an additional pruning phase. As opposed to previous prototypical-part-learning approaches, the use of super-prototypes allows ProtoSpArX to capture spatial relationships between prototypical-parts. Using an MLP for classification allows ProtoSpArX to capture non-linear relationships between super-prototypes, while applying the SpArX methodology allows explaining the classification outcome. Experiments show that ProtoSpArX outperforms state-of-the-art prototypical-part-learning approaches in terms of accuracy and the ability to model spatial relationships between prototypical-parts.

Future directions include expanding ProtoSpArX’s capabilities to encompass multi-modal data. Additionally, we will investigate the implementation of a user-model feedback loop to enhance the debugging process for super-prototypes. Further, we plan to deploy ProtoSpArX with real data, e.g. in the medical domain. Finally, we plan to explore various options for obtaining explanations from ProtoSpArX, including interactive forms thereof [16].

References

- [1] C. Wah, S. Branson, P. Welinder, P. Perona, S. Belongie, Caltech-UCSD Birds-200-2011 (CUB-200-2011), Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.
- [2] C. Rudin, Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, *Nat. Mach. Intell.* 1 (2019) 206–215. URL: <https://doi.org/10.1038/s42256-019-0048-x>. doi:10.1038/s42256-019-0048-x.
- [3] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, J. K. Su, This looks like that: deep learning for interpretable image recognition, *Advances in neural information processing systems* 32 (2019).
- [4] D. Rymarczyk, L. Struski, J. Tabor, B. Zielinski, ProtoPShare: Prototypical parts sharing for similarity discovery in interpretable image classification, in: F. Zhu, B. C. Ooi, C. Miao (Eds.), *SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, ACM, 2021, pp. 1420–1430.
- [5] M. Nauta, R. van Bree, C. Seifert, Neural prototype trees for interpretable fine-grained image recognition, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2021)*, Computer Vision Foundation / IEEE, 2021, pp. 14933–14943.
- [6] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, K. Saenko, Learning to reason: End-to-end module networks for visual question answering, in: *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 804–813. doi:10.1109/ICCV.2017.93.
- [7] H. Ayoobi, N. Potyka, F. Toni, SpArX: Sparse argumentative explanations for neural networks, in: K. Gal, A. Nowé, G. J. Nalepa, R. Fairstein, R. Radulescu (Eds.), *European Conference on Artificial Intelligence (ECAI)*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, 2023, pp. 149–156.
- [8] D. Rymarczyk, L. Struski, M. Górszczak, K. Lewandowska, J. Tabor, B. Zieliński, Interpretable image classification with differentiable prototypes assignment, in: *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XII*, Springer-Verlag, Berlin, Heidelberg, 2022, p. 351–368. URL: https://doi.org/10.1007/978-3-031-19775-8_21. doi:10.1007/978-3-031-19775-8_21.
- [9] M. Nauta, J. Schlötterer, M. van Keulen, C. Seifert, Pip-net: Patch-based intuitive prototypes for interpretable image classification, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2023)*, Computer Vision Foundation / IEEE, 2023.
- [10] A. Shrikumar, P. Greenside, A. Kundaje, Learning important features through propagating activation differences, in: *ICML*, 2017.
- [11] S. Sattarzadeh, M. Sudhakar, K. N. Plataniotis, et al., Integrated grad-cam: Sensitivity-aware visual explanation of deep convolutional networks via integrated gradient-based scoring, in: *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.
- [12] Y. Goyal, Z. Wu, J. Ernst, D. Batra, D. Parikh, S. Lee, Counterfactual visual explanations, in: *ICML*, PMLR, 2019.
- [13] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, *Advances in neural information processing systems* 30 (2017).
- [14] O. Li, H. Liu, C. Chen, C. Rudin, Deep learning for case-based reasoning through prototypes:

- A neural network that explains its predictions, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
- [15] L. Deng, The MNIST database of handwritten digit images for machine learning research, *IEEE Signal Processing Magazine* 29 (2012) 141–142.
- [16] K. Cyraś, A. Rago, E. Albini, P. Baroni, F. Toni, Argumentative XAI: A survey, in: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021*, 2021.
- [17] E. Albini, P. Lertvittayakumjorn, A. Rago, F. Toni, DAX: deep argumentative explanation for neural networks, *CoRR abs/2012.05766* (2020). URL: <https://arxiv.org/abs/2012.05766>. arXiv:2012.05766.
- [18] P. Sukpanichnant, A. Rago, P. Lertvittayakumjorn, F. Toni, Neural QBAFs: Explaining neural networks under lrp-based argumentation frameworks, in: *AIxIA 2021 - Advances in Artificial Intelligence - 20th International Conference of the Italian Association for Artificial Intelligence*, Virtual Event, December 1-3, 2021, Revised Selected Papers, volume 13196 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 429–444. URL: https://doi.org/10.1007/978-3-031-08421-8_30. doi:10.1007/978-3-031-08421-8_30.
- [19] H. Ayoobi, M. Cao, R. Verbrugge, B. Verheij, Argue to learn: Accelerated argumentation-based learning, in: *20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2021. doi:10.1109/ICMLA52953.2021.00183.
- [20] Z. Yang, A. Ishay, J. Lee, Neurasp: Embracing neural networks into answer set programming, in: C. Bessiere (Ed.), *International Joint Conference on Artificial Intelligence, (IJCAI 2020)*, ijcai.org, 2020, pp. 1755–1762.
- [21] A. Daniele, T. Campari, S. Malhotra, L. Serafini, Deep symbolic learning: Discovering symbols and rules from perceptions, in: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023*, 19th-25th August 2023, Macao, SAR, China, 2023.
- [22] N. Potyka, Interpreting neural networks as quantitative argumentation frameworks, in: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, (AAAI-21)*, 2021.
- [23] H. Ayoobi, M. Cao, R. Verbrugge, B. Verheij, Argumentation-based online incremental learning, *IEEE Trans Autom. Sci. Eng.* 19 (2022) 3419–3433. URL: <https://doi.org/10.1109/TASE.2021.3120837>. doi:10.1109/TASE.2021.3120837.
- [24] H. Ayoobi, M. Cao, R. Verbrugge, B. Verheij, Argue to learn: Accelerated argumentation-based learning, in: M. A. Wani, I. K. Sethi, W. Shi, G. Qu, D. S. Raicu, R. Jin (Eds.), *20th IEEE International Conference on Machine Learning and Applications, ICMLA 2021*, Pasadena, CA, USA, December 13-16, 2021, IEEE, 2021, pp. 1118–1123. URL: <https://doi.org/10.1109/ICMLA52953.2021.00183>. doi:10.1109/ICMLA52953.2021.00183.
- [25] H. Ayoobi, H. Kasaei, M. Cao, R. Verbrugge, B. Verheij, Explain what you see: Open-ended segmentation and recognition of occluded 3d objects, in: *IEEE International Conference on Robotics and Automation, ICRA 2023*, London, UK, May 29 - June 2, 2023, IEEE, 2023, pp. 4960–4966. URL: <https://doi.org/10.1109/ICRA48891.2023.10160927>. doi:10.1109/ICRA48891.2023.10160927.
- [26] F. Leofante, H. Ayoobi, A. Dejl, G. Freedman, D. Gorur, J. Jiang, G. Paulino-Passos, A. Rago, A. Rapberger, F. Russo, X. Yin, D. Zhang, F. Toni, Contestable AI needs computational

- argumentation, CoRR abs/2405.10729 (2024). URL: <https://doi.org/10.48550/arXiv.2405.10729>. doi:10.48550/ARXIV.2405.10729. arXiv:2405.10729.
- [27] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. URL: <http://arxiv.org/abs/1412.6980>.
- [28] J. Krause, M. Stark, J. Deng, L. Fei-Fei, 3d object representations for fine-grained categorization, in: 2013 IEEE International Conference on Computer Vision Workshops, 2013, pp. 554–561. doi:10.1109/ICCVW.2013.77.