Logical Vision: Meta-Interpretive Learning for Simple Geometrical Concepts

Wang-Zhou Dai[1], Stephen H. Muggleton[2] and Zhi-Hua Zhou[1]

[1] National Key Laboratory for Novel Software Technology, Nanjing University
[2] Department of Computing, Imperial College London

Abstract. Progress in statistical learning in recent years has enabled computers to recognize objects with near-human ability. However, recent studies have revealed particular drawbacks in current computer vision systems which suggest there exist considerable differences between the way these systems function compared with human visual cognition. Major differences are that: 1) current computer vision systems learn high-level notions directly from the low-level feature space, which makes them sensitive to low-level characteristics changing. 2) typical computer vision systems learn visual concepts discriminatively instead of encoding the knowledge necessary to produce a visual representation of the class. In this paper, we introduce a framework referred as *Logical Vision* which is demonstrated on learning visual concepts constructively and symbolically. It first constructively extracts logical facts of mid-level features, then generative Meta-Interpretive Learning technique is applied to learn high-level notions because it is capable of learning recursions, inventing predicates and so on. Owing to its symbolic representation paradigm, in our implementation, *Logical Vision* is fully implemented in Prolog apart from low-level image feature extraction primitives. Experiments are conducted on learning shapes (e.g. triangles, quadrilaterals, etc.), regular polygons and right-angle triangles. These demonstrates that learning visual concepts constructively and symbolically is effective.

1 Introduction

In the past decades, the efficiency of statistical learning enables computer vision algorithms learning from vast low-level features automatically, i.e. modeling target visual concepts by local features surrounding interest points. Recently, a statistical computer vision learning algorithm – deep neural networks (DNNs) has been achieving impressive performance on a variety of computer vision tasks [4, 5]. Although it is well known that the success of DNNs owes to its ability of automatically extracting high-level concepts, recent studies revealed some major differences between them and human visual cognition [1, 9], which exists in most of statistical computer vision learning algorithms.

For example, it is easy to produce images that are completely unrecognizable to humans, though state-of-the-art visual learning algorithms believe them to be recognizable objects with over 99% confidence [1]. This is because its discriminative learning paradigm makes some synthetic images deep within a classification region in the low-level feature space can produce high confidence predictions even though they are far from natural images in the class [1].

In other cases small perturbations to the input images, which are imperceptible to human eyes, can arbitrarily change the classifier's prediction [9]. Analysis from the authors shows that the instability is caused by classifiers' sensitivity to small changes of low-level features on input images.

In order to address these problems, in this paper we propose a constructive visual concept learning framework, called *Logical Vision*, which learns high-level visual concepts symbolically. *Logical Vision* first constructs mid-level conjectures to guide the sampling of low-level features, then uses the sampled results to revise previously constructed conjectures. With the extracted mid-level feature symbols as background knowledge, a generalized Meta-Interpretive learner [6] is used to learn high-level visual concepts because it enhances the constructive paradigm of Logic Vision through its ability to learn recursion, inventing predicates and learning from a single example. In this work, we applied our *Logical Vision* framework to tasks involving learning simple geometrical concepts such as triangles, quadrilaterals, regular polygons and so on. Owing to its symbolic representation, *Logical Vision* can be fully implemented in Prolog given low-level image feature extraction primitives as the initial background knowledge. Our experimental results show its effectiveness in learning target visual concepts which are difficult for typical low-level feature based statistical computer vision algorithms.

2 The proposed framework

In this section we introduce the framework of *Logical Vision*. The input for *Logical Vision* consists of a set of geometrical primitives B_P, one or a set of images $\mathcal{I}$ as background knowledge, and a set of logic facts E representing the examples as the target visual concepts. The task is to learn a hypothesis H that defines the target visual concept where $B_P, \mathcal{I}, H \models E$.

2.1 Constructive mid-level features extraction

The purpose of mid-level features extraction is to obtain necessary logical facts B_A representing mid-level features of $I \in \mathcal{I}$.

It is realized by repeatedly executing a "conjecturing and sampling" procedure which uses the mid-level feature conjectures to guide the sampling of low-level features. The resulting features are then used to revise previously constructed conjectures. Formally, mid-level feature extraction of an image $I \in \mathcal{I}$ is described as follows:

1. Sample low-level features F in a subarea (e.g. surrounding a focal point) of I, then add F into the sampled low-level features set $\mathcal{F}$.
2. Conjecture a mid-level feature (edge, region, texture, etc.) C according to $\mathcal{F}$.
3. Validate the conjecture C on image I by doing few more sampling. If the validation failed, reject C and go to 1, otherwise go to 4.
4. When C is valid, add it to mid-level feature set B_A, then remove the low-level features $f(C)$ that encapsulated by C, the rest of low-level features $\mathcal{F}' = \mathcal{F} - f(C)$.
5. If $\mathcal{F}' = \phi$, terminate the construction procedure and return B_A, otherwise go to 1.

Algorithm 1 $LogicalVision_{Poly}(B_P, I, Metagol_{LogicalVision}, E, N)$

Input: Geometrical primitives B_P, input image I, examples E, Meta-Interpretive learner $Metagol_{LogicalVision}$, sampling level N.
Output: Hypothesis of the target visual concept H;
Start:
Initialize edge points set $\mathcal{F} = \phi$ and sampled edges set $B_E = \phi$;
Randomly sample two edge points `P1` and `P2`, let $\mathcal{F} = \mathcal{F} \cup \{$`P1,P2`$\}$;
repeat
  Select a pair of edge points `P1,P2`$\in \mathcal{F}$;
  Validate whether `P1P2` forms an edge by querying `edge(P1,P2,`N`)`;
  if `edge/3` succeeded **then**
    Extend `P1P2` on both of its directions to form a conjecture of edge C;
    $B_E = B_E \cup C$;
    Remove all edge points `P`$\in \mathcal{F}$ that lies on edge C;
  else
    Randomly sample a line which crosses the line segment `P1P2` for new edge points, if they are not encapsulated by any sampled edge in B_E then add them into $\mathcal{F}$;
  end if
until $\mathcal{F} = \phi$;
Find connected edges in B_E to construct facts of polygons B_A;
Learn a hypothesis H with $B_A, Metagol_{LogicalVision}, B_P, E$ through MIL;
Return: H.

2.2 Meta-Interpretive Learning

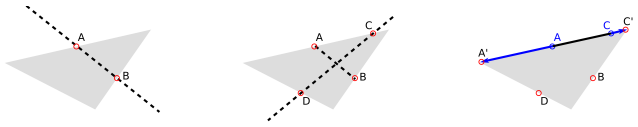
After obtaining mid-level features B_A, *Logical Vision* uses a generalized Meta-Interpretive Learner to learn target visual concepts. The input of generalized Meta-Interpretive Learning (MIL) [7] consists of a generalized Meta-Interpreter B_M and domain specific primitives B_P together with two sets of ground atoms as background knowledge B_A and examples E respectively. The output of MIL is a revised form of the background knowledge containing the original background knowledge B_A, domain specific primitives B_P augmented with additional ground atoms representing a hypothesis H.

3 Implementation

Below we describe the implementation of *Logical Vision* on the task of polygon shapes learning. The target concepts of this task are definitions of different kinds of polygons (e.g. triangles, regular polygons, etc.). Our implementation is displayed as Algorithm 1, which is referred as $LogicalVision_{Poly}$.

3.1 Polygon extraction

To learn the concepts of polygon shapes, we targeted the mid-level features B_A to be extracted as polygons. They are denoted as `polygon(Pol_i, [Edge1,...,EdgeN])`. The process of polygon extraction can be split into two stages: edge discovery and polygon construction. For simplicity, in the second stage $LogicalVision_{Poly}$ groups



(a) (b) (c)

Fig. 1. (a) 2 edge points A and B are sampled; (b) Edge AB is conjectured but it is invalid, so a random line crossing AB is sampled, 2 new edge points C and D are discovered; (c) Edge AC is conjectured and it is valid, so AC is extended until no continuous edge points were found. Finally the edge $A'C'$ is recorded and points A and C are removed from $\mathcal{F}$.

of connected edges are collected as a list. The major challenge is to discover those edges. Here we define the edge conjecture as:

```
edge(P1,P2,N):-midpoint(P1,P2,P),edge_point(P1),edge_point(P2),
                    N1 is N-1,edge(P1,P,N1),edge(P,P2,N1).
```

in which `P1` and `P2` are the conjectured end points of an edge, N is the recursion limit that controls the depth of edge validation and `midpoint/3` finds the midpoint between two points. The predicate `edge_point/1` is a primitive interacting with low-level features on images. It is true when the color gradient magnitude of pixel `P` exceeds a predefined threshold. We implemented an image-processing program by OpenCV [3], and used a C++-Prolog interface to enable communication between predicate `edge_point/1` and input images. An example of the extraction is illustrated in Fig. 1.

3.2 $Metagol_{LogicalVision}$

The polygon extraction procedure in 3.1 sometimes results in a noisy B_A (for example in Fig. 3). This causes the depth-first search in Metagol to fail or return ground clauses covering only one example. Thus, we altered the Metagol to evaluate hypotheses using foil gain [8] and preserve the best one during its hypothesis searching process.

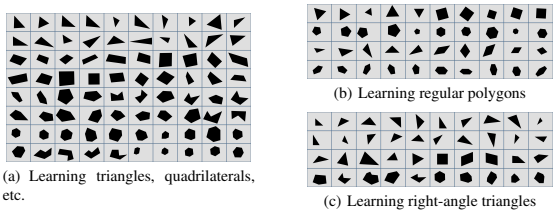
The domain specific primitives B_P of $Metagol_{LogicalVison}$ include necessary predicates for learning polygon shape related concepts. For example, `angle_list/2` returns all angles of a polygon and `std_dev_bounded/2` tests whether the standard deviation of a list of double numbers is bounded.

4 Experiments

4.1 Materials

We used Inkscape[3] to randomly generate 3 labeled image datasets for 3 polygon shape learning tasks respectively. For simplicity, the images are binary-colored, each image contains one polygon. Target concepts of the 3 task are: 1) `triangle/1`, `quadrilateral/1`, `pentagon/1` and `hexagon/1`; 2) `regular_poly/1` (regular polgon); 3) `right_tri/1` (right triangle). Note that in the third task, we used the best hypothesis of `triangle/1`

[3] http://inkscape.org

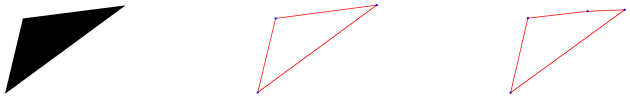


(a) Learning triangles, quadrilaterals, etc.

(b) Learning regular polygons

(c) Learning right-angle triangles

Fig. 2. Datasets for 3 learning tasks



(a) (b) (c)

Fig. 3. Noise of polygon extraction: (a) is the ground truth image. (b) and (c) are two polygons extracted by our algorithm, where (c) contains a redundant vertex.

Table 1. Result of learning simple geometrical shapes on single object datasets

	VLFeat		$LogicalVision_{Poly}$	
	Acc	F1	Acc	F1
`triangle`	0.91 ± 0.06	0.82 ± 0.09	1.00 ± 0.00	1.00 ± 0.00
`quadrilateral`	0.71 ± 0.08	0.41 ± 0.06	0.98 ± 0.03	0.96 ± 0.06
`pentagon`	0.79 ± 0.09	0.48 ± 0.24	1.00 ± 0.00	1.00 ± 0.00
`hexagon`	0.94 ± 0.04	0.85 ± 0.12	0.99 ± 0.03	0.97 ± 0.06
`regular_poly`	0.60 ± 0.10	0.72 ± 0.06	1.00 ± 0.00	1.00 ± 0.00
`right_tri`	0.75 ± 0.18	0.81 ± 0.11	1.00 ± 0.00	1.00 ± 0.00

learned in the first task as background knowledge. The datasets are presented in Fig. 2. All datasets were partitioned into 5-folds respectively, 4 of them were used for training and the remainder for testing. Thus each experiment was conducted 5 times.

4.2 Methods

$\mathbf{LogicalVision_{Poly}}$: This is the proposed approach. In order to handle the noises introduced by polygon extraction (e.g. Fig. 3), for each image we ran the extraction procedure five times independently to duplicate input instances. During evaluation, learned hypotheses were tested on all the five extracted polygons and the final prediction was based on an equal weighted vote.

VLFeat [10]: This is a popular statistical computer vision learning toolbox. The feature used by VLFeat in our experiments is PHOW [2]. It is a dense SIFT descriptor that has been widely used by current computer vision learning systems. Because the sizes of datasets are small, we used an SVM learner for VLFeat.

4.3 Results

Table 1 shows the results of our experiments. Performance of compared methods are measured by both predictive accuracy and F1-score. Clearly the performance of

$LogicalVision_{Poly}$ is significantly better than VLFeat. Following is an example of a learned hypothesis for the concept of regular polygon:

```
regular_poly_1(A,G):-angles_list(A,B),std_dev_bounded(B,G).
regular_poly_0(A,A2):-polygon(A,B),regular_poly_1(B,A2).
regular_poly(A):-regular_poly_0(A,0.02).
```

5 Conclusion

This paper studies a novel approach to the problem of visual concept learning, distinct from that employed by traditional computer vision learning algorithms. By using the proposed *Logical Vision* approach, we are able to extract logical facts of mid-level features and learn high-level visual concepts from images constructively and symbolically. The experimental results indicate that the proposed framework has potential to analyse which are traditionally hard for more statistically-oriented approaches.

In future extensions of this work we hope to compare the approach empirically with a broad set of state-of-the-art statistically-based vision algorithms. We also intend to extend our study to images involving a multiplicity of overlapping colored polygons.

References

1. Ahn, N., Yosinski, J., Clune, J.: Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In: Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA (2015)
2. Bosch, A., Zisserman, A., Muñoz, X.: Image classification using random forests and ferns. In: Proceedings of IEEE the 11th International Conference on Computer Vision. pp. 1–8. Rio de Janeiro, Brazil (2007)
3. Gary, B.: Opencv library. http://http://opencv.org/ (2000)
4. Krizhevsky, A., Sutskever, I., E. Hinton, G.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012)
5. Le, Q.V., Zou, W.Y., Yeung, S.Y., Ng, A.Y.: Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: Proceedings of 2011 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3361–3368. Colorado Springs, CO (2011)
6. Muggleton, S.H., Lin, D., Pahlavi, N., Tamaddoni-Nezhad, A.: Meta-interpretive learning: application to grammatical inference. Machine Learning 94(1), 25–49 (2014)
7. Muggleton, S.H., Lin, D., Tamaddoni-Nezhad, A.: Meta-interpretive learning of higher-order dyadic datalog: Predicate invention revisited. Machine Learning (2015), published online: DOI 10.1007/s10994-014-5471-y
8. Quinlan, J.R.: Learning logical definitions from relations. Machine Learning 5, 239–266 (1990)
9. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I.J., Fergus, R.: Intriguing properties of neural networks. CoRR abs/1312.6199 (2013)
10. Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/ (2008)