Composing Neural Learning and Symbolic Reasoning with an Application to Visual Discrimination

Anonymous Authors

Abstract

We consider the problem of combining machine learning models to perform higher-level cognitive tasks with clear specifications. We propose a novel problem of Visual Discrimination Puzzles (VDP) that requires finding interpretable discriminators that classify images according to a logical specification. Humans can solve these puzzles with ease and give robust, verifiable, and interpretable discriminators. We propose a compositional neurosymbolic framework that combines a neural network to detect objects and relationships and a symbolic learner that finds interpretable discriminators. We create a large class of VDPs ranging over natural and artificial scenes and evaluate the efficacy of our framework.

1 Introduction

Deep learning has made significant strides in solving specialized tasks, especially in areas such as vision and NLP . In this paper we study how these specialized learned models can be *composed and integrated* into solutions for higher-level tasks with clear specifications. We are especially interested in solutions that can be obtained without access to a lot of data for the higher-level task.

We believe that this problem of building mechanisms that integrate learned models to robustly solve higher-level tasks is important, and can have many applications. For example, a robot may need to formulate a complex plan to achieve a goal by utilizing a pretrained vision engine to detect objects and obstacles. It is challenging to decompose a high-level task specification into neural and mechanistic (symbolic) components such that the composed system achieves the task. In particular, we need to determine the precise interface between the components, handle the ineffectiveness and non-robustness of the neural components, and ensure the interpretability of the decisions of the entire system.

Visual Discrimination Puzzles: In this paper, we propose a new high-level task called a *Visual Discrimination Puzzle* (VDP). Figure 1 shows an example of a VDP . The first row contains some *example* images E (a, b, and c), and the second row consists of *candidate* images C (#1, #2, and #3). To solve the puzzle we must answer the following question:

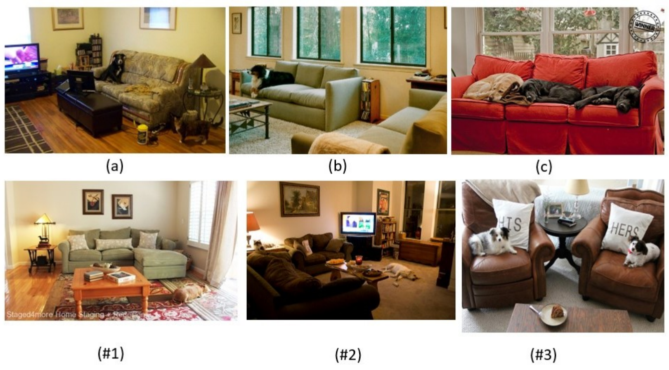

Figure 1: *A Visual Discrimination Puzzle*

Which candidate image is most similar to all the example images? Explain why.

We seek compositions of learned models for vision in order to solve VDPs without specifically training on them. Humans have no prior experience with VDPs, but seem to be able to solve the puzzles with ease. We invite the reader to try solving the puzzle in Figure 1 before reading further.

When solving VDPs, people can (and do, in our experience) come up with different answers. Many point to #3 as the answer, explaining that all example images and candidate image #3 have *all dogs sitting on couches* while the other candidate images do not. A natural formalization of the above specification of VDP puzzles is:

> *Is there a property P that is shared between all example images and one candidate image c, but is not true for the other candidate images?*

We call such a concept P a *discriminator*. Finding a discriminator leads to identifying the candidate c that is similar to the examples in a way that the other candidates are not.

Observe the following salient aspects of the problem. First, the set E of example images is small. A VDP solver must learn the common concept P using only 3-5 images. Second, unlike visual question answering (VQA) where one *answers* (or computes) a query on the scene depicted by an image, solving VDPs requires *searching* in a large space of discriminators for one that satisfies the puzzle specification.

Third, all the images in a puzzle are intricately tied in the logical specification of the problem. A solver cannot merely identify a candidate image as being most similar to all the example images— it needs to find a common concept that ties the examples together with the chosen candidate, while *excluding* the other candidates. Consequently, a candidate c may be a solution to one puzzle but to another puzzle identical in all images except one. For example, even if we changed the *non-answer* candidate image #1 in Figure 1 to have the dog on a sofa, the solution could be different from #3.

Finally, we would accept an answer from a person or machine only if they can justify it— we want to know the precise concept P that discriminates the images, be able to interpret it, and evaluate the discriminator on images to see that it indeed satisfies the puzzle specifications.

A Neuro-Symbolic Framework: We propose that effective integrative frameworks can be obtained by composing learned neural models and symbolic reasoning, where the latter caters to the higher-level task specification. Designing such a framework poses several challenges, including **(1) Interface:** determining the exact communication between the neural and symbolic components; **(2) Interpretability:** symbolic components need to explain their decisions, potentially in terms of the inputs they receive from the neural components; and **(3) Robustness:** the symbolic components should be robust to differential abilities and success of neural components.

In this paper, we instantiate a neuro-symbolic architecture for solving VDP puzzles which addresses these three challenges in novel ways. The neural models we use are state-of-the-art vision networks, trained offline using thousands of images. Given a puzzle they detect, for each image independently, a *scene graph* consisting of objects, their labels and bounding boxes or relative positions in the image, and relationships between the objects.

We propose an interface to the symbolic component using *first-order logic scene models* that can be automatically computed from scene graphs. The symbolic component uses discrete search (realized efficiently using SAT solvers) to synthesize a property P expressible in *first-order logic* over scene models and a candidate c such that P acts as a discriminator for identifying c according to the definition above. The symbolic synthesis hence not only solves the puzzle by finding an appropriate candidate, but also gives an interpretable first-order formula to justify the choice. For the puzzle in Figure 1, our system finds the discriminator $\forall x.\ (\mathsf{dog}(x) \implies \exists y.\mathsf{sofa}(y) \land \mathsf{sitting\_on}(x, y))$.

The robustness of such a system certainly depends on the robustness of the vision model (for example, it's hard to solve the puzzle in Figure 1 if a dog is not detected). However, there are other robustness properties of interest. We identify an important property called the *extension property* to ensure that discriminators do not rely too much on the level of detail at which a vision model detects objects and relationships. It ensures that if the vision layer detects additional objects or relationships, then a previously discovered discriminator continues to be a discriminator with the extended scene models.

We build a domain-specific logic for scene discriminators called *First-Order Scene Logic* (FO-SL) and prove theoretically that it has the extension property. We use FO-SL to find discriminators.

The problem of synthesizing quantified FO discriminators that classify FO models is a relatively new problem (as opposed to *program* synthesis). We engineer new effective symbolic synthesis for quantified discriminators using SAT solvers because encodings to off-the-shelf synthesizers do not scale.

Evaluation: We create 3 VDP datasets, two based on real-world scenes containing $\sim$ 9000 puzzles and a synthetic one based on the CLEVR domain containing 825 puzzles. We implement and evaluate our compositional framework on the real-world VDPs and show that it is effective and robust, solving 68% and 80% of the puzzles in the two datasets while giving interpretable discriminators. We also report ablation studies that examine the effectiveness of the interface, the FO-SL fragment and the novel synthesis algorithm. We perform the ablation by implementing a second synthesis solver based on different techniques. The CLEVR VDP puzzles are created such that they have a unique minimal discriminator in our DSL, and is used primarily to evaluate two baselines , which perform poorly in solving puzzles. We also create a dataset of 1872 puzzles with a different specification (picking the odd one out from a set of images) over the CLEVR domain to evaluate the ability of various solvers to adapt to new high-level specifications without retraining.

Contributions: The primary contributions of this paper are: (a) the new cognitive problem of VDP, which requires image understanding as well as search for interpretable discriminators, (b) a set of $\sim$ 11600 VDP and OddOne puzzles spanning both natural scenes as well as synthetic domains, (c) an instantiation of the neurosymbolic framework, with novel FO-model interface between neural networks and two symbolic engines, a robust DSL for visual discriminators and an efficient synthesis algorithm for discriminators based on SAT solving.

2 Related Work

The idea of learning in two phases, in which a first phase learns specific concepts over a large dataset and a second phase solves a few-shot learning problem by composing concepts from the first phase, is not new. The work on recognizing handwritten characters [Lake *et al.*, 2015] explores a similar idea. In this work, the first phase learns a generative model of handwritten characters using strokes and solves the second phase using Bayesian learning. In our work, we use neural models for object detection and then SAT-based symbolic learning of first-order formulae. However, our VDP puzzles are different in that we require the chosen candidate to be discriminated from other candidates, suggesting the use of a logic solver. The idea of synthesizing programs to explain behavior and generalize has been explored in various other work recently [Ellis *et al.*, 2018; Liu *et al.*, 2019]. Non-symbolic approaches to one-shot learning have also been explored [Fei-Fei *et al.*, 2006].

Synthesizing programs from discrete data has been studied by both AI and programming languages communities; the former in inductive logic programming (ILP) [Nédellec, 1998], and the latter in program synthesis [Alur *et al.*, 2018; Gulwani *et al.*, 2015] including the use of SAT/SMT solvers [Solar-Lezama *et al.*, 2006; Padhi, 2021]) .

There has been a flurry of recent work in combining neural and symbolic learning techniques [Amizadeh *et al.*, 2020; Yi *et al.*, 2018; Mao *et al.*, 2019; Wu *et al.*, 2017] that are not just

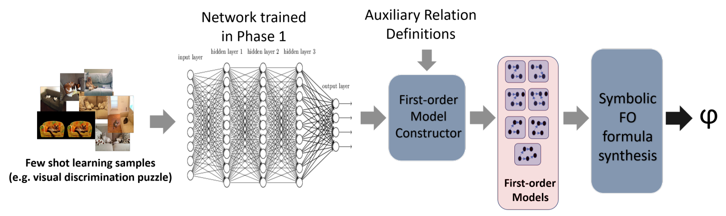
Network trained in Phase 1
Auxiliary Relation Definitions
input layer
hidden layer 1
hidden layer 2
hidden layer 3
output layer
First-order Model Constructor
First-order Models
Symbolic FO formula synthesis
φ
Few shot learning samples (e.g. visual discrimination puzzle)

Figure 2: *Learning Framework Combining Neural Learning and Symbolic Learning*

for few-shot learning problems. In some contexts, the learner needs to output a program (e.g., learning programs as models of natural scenes [Liu *et al.*, 2019], helping programmers write code by learning from large code repositories [Raychev *et al.*, 2019]), calling for combining neural and symbolic learning, and several new techniques have emerged [Balog *et al.*, 2017; Gaunt *et al.*, 2017; Valkov *et al.*, 2018; Parisotto *et al.*, 2017; Murali *et al.*, 2018; Devlin *et al.*, 2017; Bunel *et al.*, 2018; Sun *et al.*, 2018; Evans and Grefenstette, 2018]. In this context, our work is novel in that it combines the neural and symbolic parts in two different layers, where the symbolic layer is used to synthesize interpretable logical discriminators and handle few-shot learning effectively.

A closely related problem is that of Visual Question Answering (VQA) [Antol *et al.*, 2015], with neuro-symbolic approaches [Yi *et al.*, 2018; Amizadeh *et al.*, 2020] that disentangle visual and NLP capabilities from reasoning to solve VQA on artificially rendered images. VQA involves *queries* about a single image/scene and end-to-end learning algorithms are commonly used.However, the VDP puzzles require *searching* through a large class of potential discriminators to solve a puzzle, which is inherently a higher-level task. In a sense, we are asking whether there is *some question* for which the answers to these questions on the various images acts as a discriminator. The work in [Andreas *et al.*, 2017] solves few-shot classification problems in this manner, but cannot handle specifications like VDPs involving negation.

Another related problem is Raven's Progressive Matrices [Santoro *et al.*, 2018; Zhang *et al.*, 2019]. While the puzzles are similar to ours in specification, the space of concepts is minuscule and does not require any synthesis. The work in [Zhang *et al.*, 2019] shows that simply enumerating the concepts achieves 100% accuracy on these datasets.

3 Composing Neural Learning and Symbolic Reasoning for Discriminating Scenes

In our experience with people solving VDPs, humans have first learned to distinguish objects (*dogs, sofas*, detect relationships (*dog sitting on sofa*), poses (*woman standing*), and other attributes (*cat has closed eyes*) using a rich visual experience accumulated from childhood. When they are proposed a VDP puzzle with the task specification, they do not have the same rich experience to go by (they likely haven't solved any VDPs previously). However, they can quickly formulate a mechanism that identifies a common concept which ties together the few images in the puzzle by *composing high-level concepts built using pre-learned lower-level concepts*.

Our framework is built on the above intuition and has two similar phases. We propose a neuro-symbolic learning technique to solve VDPs, as illustrated in Figure 2 . Phase 1 involves *long-term learning* using large training sets and is independent of the notion of any high-level task. Phase 2 is engineered for the specification of the particular task (solving a VDP), and utilizes the concepts learned in Phase 1 to achieve the higher-level specification (few-shot discrimination of scenes). More precisely, we propose:

Neural Learning Algorithms for Phase 1: Learning scene representations in terms of objects ("dog"), attributes ("dog is black"), and relationships ("dog is sitting on sofa") — called a scene graph — is a well-studied problem in literature, and deep CNNs result in effective models. In this paper we use YOLOV4 [Bochkovskiy *et al.*, 2020], a CNN-based object detector trained on IMAGENET and COCO datasets to predict multiple objects with *bounding boxes* and *class labels* .

Interfacing Phase 1 and Phase 2 using FO Scene Models: The interface between the outputs of the neural network and the symbolic synthesis module is an important challenge. We propose a novel interface, namely *First-Order Scene Models* that capture object classes, attributes, and relationships.

Symbolic Logic Learning Algorithms for Phase 2: In this phase, we take the *finite* first-order scene models for each image and build novel symbolic synthesis algorithms for *quantified first-order logic* formulas that discriminate between these models, respecting the higher-level puzzle specification.

If solving VDPs is the only goal, we could train models on a large class of puzzles. However, we focus on a different kind of solution in this paper; we want to study how to solve puzzles/tasks when presented afresh, without access to a rich experience of solving them.

4 Symbolic Synthesis of First-Order Logic Discriminators from FO Models

4.1 First-Order Logic and FO Models for Scenes

We work with first-order *relational* logic over a signature $\Sigma = (\mathcal{L}, \mathcal{R})$, where $\mathcal{R}$ is a finite set of relation symbols and $\mathcal{L} \subseteq \mathcal{R}$ is a set of unary symbols we call *labels* (which we use to model categories of objects). Each relation symbol is associated with an arity $n \in \mathbb{N}, n > 0$.

The syntax of first-order logic formulas is given by:

$$\textit{Formulas } \varphi ::= R(x_1, \ldots, x_k) \mid x_i = x_j \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \neg\varphi \mid \varphi \Rightarrow \varphi \mid \exists x.\, \varphi \mid \forall x.\, \varphi$$

Models and semantics for first-order logic are the usual ones (see a standard logic textbook).

Given a set of images X for a VDP puzzle, we build a first-order model for each image $I \in X$ by feeding I to the pretrained neural network. The model's universe corresponds to the set of objects detected by the network. We model the class labels identified by the network as unary predicates $\mathcal{L}$ (e.g. $cat(x)$ or $person(x)$), and the identified relationships between objects as relations $\mathcal{R}$. The resulting *First-Order Scene Models* are used by the symbolic synthesizer.

4.2 FO Scene Logic and the Extension Property

We argue that a first-order logic for scenes obtained from neural network detection needs to satisfy a particular *robustness property* that we would like discriminators to have — that discriminators remain discriminators when *additional* objects or relationships are discovered by a vision model.

Suppose that we have found a discriminator for a puzzle, say, *all dogs are on sofas*. We would like it to remain a discriminator if new *irrelevant* objects and relationships are detected and added to the scene model. For example, adding dogs/ whiskers or a previously unrecognized pen on the sofa to the scene model should not make the discriminator wrong.

Standard first-order logic does not have this property. For example, in the puzzle in Figure 1, if only dogs and sofas were recognized we could express "all dogs are on a sofas" using the formula $\forall x.\exists y.(\mathit{sofa}(y) \wedge (x \neq y \Rightarrow \mathit{on}(x, y)))$ which says that all objects other than a sofa are on a sofa. If the vision model now recognizes tables in the images, then the formula fails to be a discriminator (false on example images).

We define *model extensions* as follows:

Definition (Model Extension) A model M' over $\mathcal{R}'$ *extends* a model M over $\mathcal{R}$ (where $\mathcal{R} \subseteq \mathcal{R}'$) if M' agrees with M on the interpretation of all relations in $\mathcal{R}$.

Thus, given (i) the imprecision of vision models, (ii) that different systems will likely have different detection rates for various object classes/relationships, and (iii) that choice of visual system crucially affects the scene model and consequently the discriminators, we propose the following property for discriminators:

Definition (Extension Property) A logic *has the extension property* if any discriminator φ in the logic for a set of models $\{M_i\}$ remains a discriminator for any extended models $\{M_i'\}$.

We formulate *First-order Scene Logic* (FO-SL) based on guarded logics, where quantified objects are always *guarded* by an assertion that they have a specific object label:

$$\text{FO–SL} ::= \forall x.L(x) \Rightarrow \varphi \mid \exists x.L(x) \wedge \varphi \mid \psi$$
$$\psi ::= R(\overline{x}) \mid \psi \vee \psi \mid \psi \wedge \psi \mid \neg\psi \mid \psi \Rightarrow \psi$$

where L ranges over label relations, e.g. $\mathit{cat}(x)$, and R ranges over all relation symbols (attributes and object relationships).

FO-SL can express properties of all cats in a scene, but not of all objects of *any* kind (a variable cannot range over cats, pens, paintings, and specks of dust). With this we can show (see Appendix for proof):

Theorem. The guarded fragment has the extension property. We thus propose the use of FO-SL to learn discriminators.

4.3 Solving VDP using Formula Synthesis

We formulate the problem of solving a VDP as the problem of synthesizing a formula that acts as the discriminator between FO scene models. Let us fix a puzzle with example E and candidate C images, and signature Σ as determined by a vision model. Let the corresponding FO scene models be $E_M = \{e_1^M, \ldots, e_u^M\}$ and $C_M = \{c_1^M, \ldots, c_v^M\}$.

Definition (Discriminator) An FO-sentence φ is a *discriminator* for (E_M, C_M) if there is a model $\widehat{c}^M \in C_M$ such that:
(D1) For every model $e^M \in E_M$, $e^M \models \varphi$
(D2) $\widehat{c}^M \models \varphi$
(D3) For every $c^M \in C_M$ such that $c^M \neq \widehat{c}^M$, $c^M \not\models \varphi$

This definition formally captures the puzzle specification: a discriminator is a formula that satisfies all example images (D1) and *precisely* one of the candidate images (D2 and D3).

Learning FO-SL Discriminators We describe algorithms to synthesize FO-SL discriminators to solve a given VDP puzzle. In particular, we formulate the synthesizer we use to learn *conjunctive* discriminators.

First, we can encode the problem to state-of-the-art *program synthesis* engines handling the SyGuS format (Syntax-Guided Synthesis) [Padhi, 2021], but they did not scale well (see Section 5.3). We formulate a new solution for synthesizing quantified FO formulae using SAT solvers.

We fix k, the number of quantifiers in the discriminator. k is initially set to 1, and incremented iteratively whenever we cannot find a discriminator with k quantifiers. We now introduce a Boolean variable b_a for every atomic formula $a(\overline{x})$ that can occur in the matrix (inner quantifier-free portion) of the formula. The intention is that b_a is true if and only if the atomic formula a occurs as a conjunct in the matrix of the discriminator. We also introduce k Boolean variables that choose whether the k quantifiers are existential or universal quantifiers, and some more Boolean variables that choose the guards (labels in FO-SL) for each quantified variable. Given a valuation over these Booleans $\overline{b}$, we can write a formula $\phi(\overline{b})$ that evaluates the discriminator encoded by $\overline{b}$ on all images. With extra Boolean variables that encode candidate choice, we can formulate a constraint that models the specification of the puzzle, as in the discriminator definition above.

We can then ask a SAT solver (such as Z3 [De Moura and Bjørner, 2008]) whether the constraint is satisfiable. If the SAT solver finds a valuation, we can construct the FO-SL discriminator and the chosen candidate from it. If the SAT solver says the constraint is unsatisfiable, we know there is no conjunctive discriminator with k quantifiers.

5 Evaluation

5.1 Datasets

We create 11,600 puzzles across four datasets. We describe these briefly; see Appendix for details. We also invite the reader to browse the static **website of VDPs** provided in the Appendix for a sample of puzzles across the datasets.

Natural Scenes The Natural Scenes VDP dataset is created from 20 *base* real-world concept classes such as 'All dogs are on sofas'. For each class, we collect positive images that satisfy the concept and negative images that do not. We create puzzles by choosing all examples from the positive set and all candidates from the negative set except for one positive image (the *intended* candidate), and sample 3864 random puzzles. We provide a description of these concepts in Table 1.

GQA VDP dataset The GQA VDP dataset is created automatically using the GQA dataset [Hudson and Manning, 2019], which is a VQA dataset. It consists of real-world scenes along with scene graphs, as well as questions and answers about the images. We use questions with yes/no answers such as *Is there a fence made of wood?* and create puzzles as above (with 'yes' images being the positive category), sampling 5000 random VDPs. Note that the proposition *There is a fence made of wood* in the question is hence a discriminator.

CLEVR VDP dataset The CLEVR domain [Johnson *et al.*, 2017] is an artificial VQA domain consisting of images with 3D shapes such as spheres, cubes, etc. The objects possess

ID	Class Description	ID	Class Description
1	All teddy bears on a sofa	2	There is an SUV
3	Onward lane (pedestrian on left of street)	4	Fruit in separate piles (apples and oranges)
5	Kickoff position (football b/w two people)	6	Laid out place setting (v/s dirty dishes)
7	Person kicking ball	8	Dog herding sheep
9	Parking spot	10	People carrying umbrellas
11	Bus filled with people	12	All dogs on sofas
13	Desktop PC	14	People wearing ties
15	Person sleeping on bench	16	All cats on sofas
17	Kitchen	18	TV is switched on
19	Two cats on same sofa	20	Cat displayed on TV

Table 1: Concept Class Descriptions for Natural Scenes Dataset

ID	Concept Class Schema
1	Every `shapeX` has a `shapeY` to its left and right.
2	There is a `shapeX` of color `colorA` to the left of a `shapeY` of color `colorB`.
3	There is a `shapeX` to the right of every `shapeY`.
4	There is a `shapeX` and there is a `shapeY` to the left of all `shapeX`.
5	Every `shapeX` has a `shapeY` to its right.
6	All `shapeX`s and `shapeY`s have the same color.
7	There is no color such that there is only one `shapeX` of that color.
8	There is a leftmost `shapeX` and a rightmost `shapeX`.
9	All `shapeX` are to the left of all `shapeY`s and the rightmost `shapeY` is made of `materialQ`.
10	All `shapeX`s are to the left of a `shapeY` of color `colorB`.
11	All `shapeX`s are to the left of all `shapeY`s.
12	Every `shapeX` has a `shapeY` to its right.
13	Every `shapeX` has a `shapeY` behind it.
14	There are three `shapeX`s of the same color.
15	There is a `materialQ` `shapeX` to the left of all `shapeY`s.

Table 2: Concept Class Schema for CLEVR VDP Dataset. `shapeX` and `shapeY` range over {sphere, cylinder, cube}, `colorA` and `colorB` over 8 possible colors, and `materialQ` over {rubber, metal}.

a rich combination of attributes such as shape, colour, size, and material. The CLEVR VDP dataset consists of 15 base concept classes as described in Table 2. Each concept class is a *schema* such as 'ShapeX and ShapeY have the same color', where variables ShapeX and ShapeY denote distinct shapes. We create abstract VDPs whose unique minimal discriminator is the FO-SL schema (eg: $\forall x.ShapeX \Rightarrow \forall y.ShapeY \Rightarrow samecolor(x, y)$). We instantiate these variables with various combinations of shapes, colours, etc. and sample 825 VDPs with unique solutions in the FO-SL conjunctive fragment.

CLEVR OddOne Puzzles dataset We create a dataset of discrimination puzzles that are different from VDPs. An OddOne puzzle consists of 4 images with the objective of identifying the *odd* image. We formalize this task similar to VDP, demanding a concept φ that satisfies *exactly* three images. We create these from our CLEVR VDP dataset by choosing three example images and one non-answer candidate image from each puzzle. Finally, we only include the 1872 puzzles that have a unique minimal discriminator in FO-SL.

Note that only the CLEVR VDP and OddOne datasets have unique discriminators in conjunctive FO-SL, not the other two.

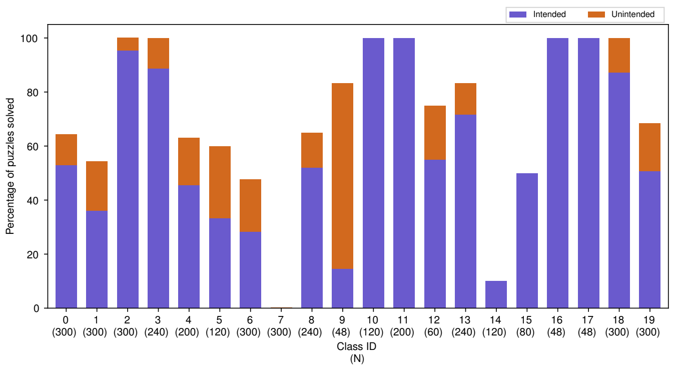

Figure 3: Evaluation on 3864 Natural Scenes Dataset puzzles: Class ID refers to the ID in Table 1, and (N) is the number of puzzles in each class. **Intended**: solver chose the intended discriminator. **Unintended**: solver did not choose the intended discriminator.

5.2 Implementation

We use a pretrained model of YOLOv4 [Bochkovskiy *et al.*, 2020] for the Natural Scenes dataset, which outputs object labels and bounding boxes. For the CLEVR domain, we use a pretrained model from the work in [Yi *et al.*, 2018] . The model produces a set of objects with their coordinates in 3D space and other attributes like shape, color, or material. We compute FO scene models based on the outputs automatically.

The symbolic synthesis engine implements an algorithm that searches for conjunctive discriminators in FO-SL. The algorithm creates the constraints described in Section 4.3 and uses the SAT-solver Z3 [De Moura and Bjørner, 2008] .We also implement another synthesis engine that searches for discriminators in full first-order logic to perform ablation studies.

See Appendix for more details about implementation.

5.3 Experiments

We report on the evaluation of our tool on the datasets and various ablation studies in terms of Research Questions (RQs).

Experiments on Natural Scenes and GQA VDP Datasets

RQ1: Effectiveness We evaluate our tool on the Natural Scenes dataset, applying the symbolic synthesis engine with a timeout. Since it is always possible to discriminate a finite set of models with a complex (but perhaps unnatural) formula, we restrict our search to discriminators with complexity smaller than or equal to the target discriminator. We present the results in Figure 3. Our tool is effective and solves 68% of the 3864 puzzles, with an average accuracy of 71% per concept class.

Our tool finds varied discriminators with multiple quantifiers and complex nesting. For example, 'Football in between two people' (kickoff position) is expressed by $\forall x.\,(\mathsf{person}(x) \Rightarrow \forall y.\,(\mathsf{sports\ ball}(y) \Rightarrow \exists z.\,(\mathsf{person}(z) \wedge \mathsf{left}(y, x) \wedge \mathsf{right}(y, z))))$ which has three quantifiers with alternation. The tool also generalizes intended discriminators.

Note that the Natural Scenes VDPs do not have unique solutions since since smaller discriminators may exist. For example, in the concept class of 'Laid out place settings', a particular puzzle always displayed cups in example images. Our tool picked an unintended candidate satisfying 'There is a cup' rather than the intended candidate satisfying a more complex discriminator (utensil arrangements). Our tool picks an unintended candidate in 17% of solutions. We provide more examples of discriminators found in Table 3 (Appendix).

RQ2: Generality by Ablating Vision Model Errors How general is our framework? The GQA VDP dataset consists of automatically generated puzzles where neither the images nor the discriminators are curated in this work. However, the pipeline fails if the vision model fails. In fact most failures on the Natural Scenes dataset are of this nature. In our experience, although vision systems are good at detecting objects, many report several bounding boxes for the same object and are not accurate. We experimented with SOTA scene graph generators and found many issues.

To better study the generality of our formulation, we ablate the vision model errors and use the ground-truth scene graphs of images given in the GQA dataset to extract scene models. Our tool performs very well, solving 81% of the puzzles (4% unintended). See Appendix for examples of (un)solved puzzles.

RQ3: Failure Modes The primary failure mode of the tool is failure of the vision model, as discussed above. This failure manifests as nonsensical discriminators. We leave the problem of designing frameworks that are robust despite vision model failures to future work.

We identify two other failure modes where the solver was not able to find any discriminator, namely (1) Expressive power of the interface: we do not solve puzzles with discriminators like *There is a bag in the bottom portion of the image*, since information about the region of the image in which an object was present is not typically expressed in scene graphs. Many of the unsolved puzzles in the GQA VDP dataset belong to this failure mode; and (2) Expressive power of FO-SL fragment: consider the concept class *TV is switched on* in Natural Scenes dataset, expressed as $\exists x.\, \mathsf{tv}(x) \wedge \exists y.\, \mathsf{displayed\ on}(y, x)$. This requires non-guarded quantification and cannot be expressed in FO-SL. Another example is *There is a dog that is not white*, which requires negation and is not expressible in the conjunctive fragment.

RQ4: Ablation of Synthesis Algorithm Our novel synthesis algorithm based on SAT is extremely effective and finds discriminators in a few seconds. However, there are engines developed in the program synthesis literature for SyGuS problems [Padhi, 2021] . The specification of a discriminator can be encoded as a SyGuS problem, and we perform an ablation study using CVC4SY [Reynolds *et al.*, 2019] (winner of SyGuS competitions) for finding discriminators. CVC4SY did not scale at all and took at least 10 minutes for even moderately difficult puzzles, often not terminating after 30 minutes.

RQ5: Ablation of FO-SL Evaluation of RQ1 shows that FO-SL is a rich logic that expresses many interesting discriminators. However, would a more expressive logic find more natural or better discriminators? We perform an ablation study on Natural Scenes VDPs by implementing a second solver for symbolic synthesis. This solver searches for discriminators in full First-Order Logic rather than FO-SL: the quantifiers are not guarded and the matrix need not be conjunctive. Note that FOL does not satisfy the extension property.

This solver is slower and times out for 50% of the puzzles. Among solved puzzles, solutions are sometimes more general e.g., 'There is something that is within all sofas', instead of *All dogs on sofas*. However, we almost always obtain unnatural

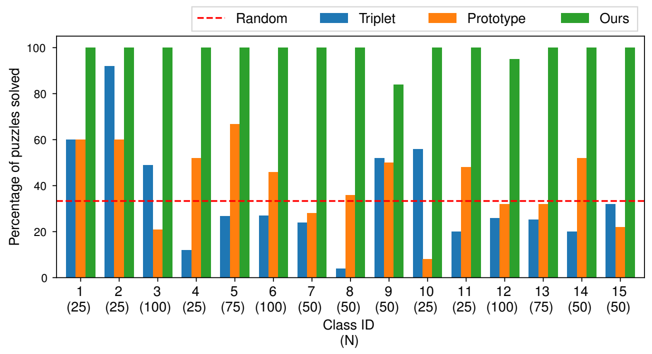

Figure 4: Comparison with neural baselines on 825 CLEVR VDP Dataset puzzles: Class ID refers to the ID in Table 2, and (N) is the number of puzzles in each class. **Triplet**: image similarity baseline. **Prototype**: prototypical network baseline. **Ours**: our solver. Dashed line represents accuracy of a random predictor.

discriminators e.g., 'There is no chair and there is some non-sofa'. We therefore conclude that the guarded quantification of FO-SL is important, and full FOL is not good for finding natural discriminators. See Appendix for more examples of unnatural discriminators (Table 4) and details of the solver.

Experiments on CLEVR VDP Dataset

RQ6: Neural Baselines The CLEVR VDP dataset consists of puzzles with unique discriminators by construction, and our tool performs unsurprisingly well (99%). Since solutions are unique, we can ask if solving VDPs is learnable using DNNs.

We first construct a baseline from an image similarity model trained using triplet loss [Wang *et al.*, 2014]. To solve a VDP, we choose the candidate that maximizes the product of similarity scores with all the example images. We present the class-wise performance of this model in Figure 4. It does not perform very well and is rarely better than random (33%). This shows that VDP solutions are not aligned well with commonly learned image features. Therefore, we evaluate another baseline based on prototypical networks [Snell *et al.*, 2017] by training on VDPs. Such networks aim to learn embeddings such that the average embedding of a class of images acts as a prototype, which can then use to compute similarity with respect to the class. We fine-tune a ResNet18 [He *et al.*, 2016] + MLP architecture pretrained on CIFAR10 using 6 concept classes, validated against a held-out set of classes. We then evaluate it on the unseen classes and other unseen puzzles. This model achieves a slightly better overall accuracy of 40% as shown in Figure 4, but is still not performant.

Experiments on CLEVR OddOne Puzzles Dataset

RQ7: Adaptive Mechanisms We evaluate the adaptability of mechanisms (without retraining) when the higher-level puzzle description is changed. Our framework is evidently highly adaptable and we can solve OddOne puzzles without retraining by simply changing the synthesis objective (the constraint).

We evaluate the adaptability of the baseline models learned for VDP on the new task by adapting the scoring function (see Appendix) and find that they perform very poorly. The similarity and prototypical network baselines perform at 13% and 23% respectively, compared to a random predictor at 25%. Therefore, we conclude that the neural representations learned using the two baselines for one task do not lend themselves well to newer high-level task specifications without retraining.

References

[Alur *et al.*, 2018] Rajeev Alur, Rishabh Singh, Dana Fisman, and Armando Solar-Lezama. Search-based program synthesis. *Commun. ACM*, 61(12):84–93, November 2018.

[Amizadeh *et al.*, 2020] Saeed Amizadeh, Hamid Palangi, Oleksandr Polozov, Yichen Huang, and Kazuhito Koishida. Neuro-symbolic visual reasoning: Disentangling "visual" from "reasoning". In *ICML 2020*, July 2020.

[Andreas *et al.*, 2017] Jacob Andreas, Dan Klein, and Sergey Levine. Learning with latent language. *CoRR*, abs/1711.00482, 2017.

[Antol *et al.*, 2015] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. *ICCV*, 2015.

[Balog *et al.*, 2017] Matej Balog, Alexander L. Gaunt, Marc Brockschmidt, Sebastian Nowozin, and Daniel Tarlow. Deepcoder: Learning to write programs. *ICLR*, 2017.

[Bochkovskiy *et al.*, 2020] Alexey Bochkovskiy, Chien-Yao Wang, and H. Liao. Yolov4: Optimal speed and accuracy of object detection. *ArXiv*, abs/2004.10934, 2020.

[Bunel *et al.*, 2018] Rudy Bunel, Matthew J. Hausknecht, Jacob Devlin, Rishabh Singh, and Pushmeet Kohli. Leveraging grammar and reinforcement learning for neural program synthesis. *ICLR*, 2018.

[De Moura and Bjørner, 2008] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. In *Proceedings of the 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, TACAS'08, pages 337–340, Berlin, Heidelberg, 2008. Springer-Verlag.

[Devlin *et al.*, 2017] Jacob Devlin, Jonathan Uesato, Surya Bhupatiraju, Rishabh Singh, Abdel-rahman Mohamed, and Pushmeet Kohli. Robustfill: Neural program learning under noisy i/o. *ICML*, 2017.

[Ellis *et al.*, 2018] Kevin Ellis, Daniel Ritchie, Armando Solar-Lezama, and Joshua B. Tenenbaum. Learning to infer graphics programs from hand-drawn images. *NIPS*, 2018.

[Evans and Grefenstette, 2018] Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *J. Artif. Int. Res.*, 61(1):1–64, jan 2018.

[Fei-Fei *et al.*, 2006] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(4):594–611, April 2006.

[Gaunt *et al.*, 2017] Alexander Gaunt, Marc Brockschmidt, Nate Kushman, and Daniel Tarlow. Differentiable programs with neural libraries. In *Proceedings of ICML'17*, July 2017.

[Gulwani *et al.*, 2015] Sumit Gulwani, José Hernández-Orallo, Emanuel Kitzelmann, Stephen H. Muggleton, Ute Schmid, and Benjamin Zorn. Inductive programming meets the real world. *Commun. ACM*, 58(11):90–99, October 2015.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016.

[Hudson and Manning, 2019] Drew A. Hudson and Christopher D. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *CVPR*, 2019.

[Johnson *et al.*, 2017] J. Johnson, B. Hariharan, Laurens van der Maaten, Li Fei-Fei, C. L. Zitnick, and Ross B. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *CVPR*, 2017.

[Lake *et al.*, 2015] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[Liu *et al.*, 2019] Yunchao Liu, Jiajun Wu, Zheng Wu, Daniel Ritchie, William T. Freeman, and Joshua B. Tenenbaum. Learning to describe scenes with programs. *ICLR*, 2019.

[Mao *et al.*, 2019] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. The Neuro-Symbolic Concept Learner: Interpreting Scenes, Words, and Sentences From Natural Supervision. *ICLR*, 2019.

[Murali *et al.*, 2018] Vijayaraghavan Murali, Letao Qi, Swarat Chaudhuri, and Chris Jermaine. Neural sketch learning for conditional program generation. *ICLR*, 2018.

[Nédellec, 1998] Claire Nédellec. Inductive logic programming, from machine learning to software engineering by f. bergadano and d. gunetti, the mit press, usa, 1996. *Knowl. Eng. Rev.*, 13(2):201–208, jul 1998.

[Padhi, 2021] Saswat Padhi. Sygus: https://sygus.org/, 2021.

[Parisotto *et al.*, 2017] Emilio Parisotto, Abdel rahman Mohamed, Rishabh Singh, Lihong Li, Dengyong Zhou, and Pushmeet Kohli. Neuro-symbolic program synthesis. *CoRR*, abs/1611.01855, 2017.

[Raychev *et al.*, 2019] Veselin Raychev, Martin T. Vechev, and Andreas Krause. Predicting program properties from 'big code'. *Commun. ACM*, 62(3):99–107, 2019.

[Reynolds *et al.*, 2019] Andrew Reynolds, Haniel Barbosa, Andres Nötzli, Clark Barrett, and Cesare Tinelli. cvc4sy: Smart and fast term enumeration for syntax-guided synthesis. *Computer Aided Verification*, 2019.

[Santoro *et al.*, 2018] Adam Santoro, Felix Hill, David G. T. Barrett, Ari S. Morcos, and Timothy P. Lillicrap. Measuring abstract reasoning in neural networks. *ICML*, 2018.

[Snell *et al.*, 2017] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *NIPS*, 2017.

[Solar-Lezama *et al.*, 2006] Armando Solar-Lezama, Liviu Tancau, Rastislav Bodik, Sanjit Seshia, and Vijay Saraswat. Combinatorial sketching for finite programs. *SIGOPS Oper. Syst. Rev.*, 2006.

[Sun *et al.*, 2018] Shao-Hua Sun, Hyeonwoo Noh, Sriram Somasundaram, and Joseph Lim. Neural program synthesis from diverse demonstration videos. *ICML*, 2018.

[Valkov *et al.*, 2018] Lazar Valkov, Dipak Chaudhari, Akash Srivastava, Charles Sutton, and Swarat Chaudhuri. Houdini: Lifelong learning as program synthesis. *NIPS*, 2018.

[Wang *et al.*, 2014] Jiang Wang, Yang song, Thomas Leung, Chuck Rosenberg, Jinbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. *CoRR*, abs/1404.4661, 2014.

[Wu *et al.*, 2017] Jiajun Wu, Joshua B Tenenbaum, and Pushmeet Kohli. Neural scene de-rendering. *CVPR*, 2017.

[Yi *et al.*, 2018] Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B. Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *NeurIPS*, 2018.

[Zhang *et al.*, 2019] Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. Raven: A dataset for relational and analogical visual reasoning. *CVPR*, 2019.