Non-monotonic Logical Reasoning and Deep Learning for Explainable Visual Question Answering

Heather Riley
Electrical and Computer Engineering
The University of Auckland, New Zealand
hril230@aucklanduni.ac.nz

Mohan Sridharan
School of Computer Science
University of Birmingham, United Kingdom
m.sridharan@bham.ac.uk

ABSTRACT

State of the art visual question answering (VQA) methods rely heavily on deep network architectures. These methods require a large labeled dataset for training, which is not available in many domains. Also, it is difficult to explain the working of deep networks learned from such datasets. Towards addressing these limitations, this paper describes an architecture inspired by research in cognitive systems that integrates commonsense logical reasoning with deep learning algorithms. In the context of answering explanatory questions about scenes and the underlying classification problems, the architecture uses deep networks for processing images and for generating answers to queries. Between these deep networks, it embeds components for non-monotonic logical reasoning with incomplete commonsense domain knowledge and for decision tree induction. Experimental results show that this architecture outperforms an architecture based only on deep networks when the training dataset is small, provides comparable performance on larger datasets, and provides intuitive answers to explanatory questions.

CCS CONCEPTS

• **Computing methodologies** → *Nonmonotonic, default reasoning and belief revision*; *Logic programming and answer set programming*; *Scene understanding*; *Neural networks*;

KEYWORDS

Non-monotonic logical reasoning, deep networks, explainable visual query answering

ACM Reference Format:
Heather Riley and Mohan Sridharan. 2018. Non-monotonic Logical Reasoning and Deep Learning for Explainable Visual Question Answering. In *6th International Conference on Human-Agent Interaction (HAI '18), December 15–18, 2018, Southampton, United Kingdom.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3284432.3284456

1 INTRODUCTION

Sophisticated algorithms developed for fundamental computer vision problems such as face recognition, gesture recognition, and obstacle avoidance, are increasingly being used in critical application domains such as healthcare and autonomous navigation. Consider an autonomous car driving on a busy road. Any error made by the car's computational system, e.g., in recognizing traffic signs, can result in serious accidents and decrease the level of trust of humans in autonomous cars. In general, it is likely that humans interacting with an autonomous system designed for complex domains will want to know why and how the system arrived at particular conclusions; this "explainability" will be especially important as humans build trust in the operation of such systems. Understanding the operation of these systems will also help human designers improve the performance of these systems. Despite considerable progress in recent years, explainable learning and decision making in autonomous systems continue to be open problems.

Visual question answering (VQA) is an example of a complex task that inherently requires explainable learning and reasoning. Given an image of a scene and a natural language question about the image as inputs, the desired output is an accurate answer to the question. In this paper, we focus on answering explanatory questions about images of input scenes and an underlying classification problem. For instance, a system recognizing traffic signs may be posed questions such as "what does this traffic sign mean?", or "how should a driver respond to this sign?", and a system estimating the stability of configurations of blocks on a tabletop, may be asked "why is this structure unstable?" or "what would make the structure stable?". We assume that these questions have been transcribed into text, and that answers to questions will also be provided as text.

Deep networks (arguably) represent the state of the art for VQA, and for many perception and control problems in which their performance often rivals that of human experts. However, these deep networks are computationally expensive to train, and require a large number of labeled training samples to learn an accurate mapping between input and output in complex domains. It is not always possible to satisfy these requirements, especially in dynamic domains where the desired mapping may change over time. Furthermore, the use of deep networks makes it more challenging to explain the observed performance of the corresponding system. Research in cognitive systems, on the other hand, indicates that explainable reasoning can be achieved while solving complex problems by reasoning with commonsense (prior) knowledge and learning from experience. Inspired by this research, the architecture described in this paper exploits the complementary strengths of reasoning with commonsense domain knowledge, inductive reasoning, and deep learning, to address the limitations of deep network architectures. It has the following components:

- Convolutional Neural Networks (CNNs) extract concise visual features from input images of scenes of interest.

HAI '18, December 15–18, 2018, Southampton, United Kingdom

ACM ISBN 978-1-4503-5953-5/18/12…$15.00
https://doi.org/10.1145/3284432.3284456

- Non-monotonic logical reasoning with the extracted features and incomplete, commonsense domain knowledge is used for classification and for answering explanatory questions.
- Feature vectors not classified by commonsense reasoning are used to train a decision tree classifier whose output, along with the visual features, train a Recurrent Neural Network (RNN) to answer explanatory questions.

We illustrate and evaluate our architecture in the context of two domains: (i) estimating the stability of configurations of simulated blocks on a tabletop; and (ii) recognizing different traffic signs in a benchmark dataset of images. Experimental results show that our architecture outperforms an architecture based only on deep networks when the training dataset is small, provides comparable performance on larger datasets, and provides intuitive explanations for both the classification output and answers to queries.

2 RELATED WORK

Current state of the art approaches for VQA are based on deep learning [9, 14–16, 27]. These approaches train neural network architectures with different configurations of layers (e.g., convolution, pooling layers) on labeled data, to capture the mapping between the input(s) (e.g., images, verbal input) and the desired output(s) (e.g., class labels or explanations). Although deep networks have demonstrated the ability to model complex non-linear mappings between inputs and outputs for different pattern recognition tasks, they are computationally expensive and require large training datasets. They also make it rather difficult to understand the internal representations, identify changes that will improve the performance of the deep networks, or to transfer knowledge acquired in one domain to other related domains. It is also challenging to accurately measure performance or identify dataset bias, e.g., deep networks have been shown to answer questions about images based on similar question-answer (training) samples without reasoning about the images [8, 21, 27]. There is on-going research to address each of these problems, e.g., to actively reduce bias in training data or reduce the amount of training data to prevent overfitting [1, 7]. However, enabling efficient learning and accurate explanations with deep network architectures continues to be an open problem.

2.1 Explainability

Researchers have developed models to understand the internal reasoning of deep neural networks. One recent approach uses the gradient in the last convolutional layer of a convolutional neural network (CNN) to compute the relative contribution (or weight) of each neuron to the classification decision made [18]. Although this approach helps explore the internal representation, the relative weights of neurons do not provide an intuitive explanation of the CNN's operation. Researchers have also developed general approaches for understanding the predictions of any given machine learning algorithm [10, 17]. They obtain an explanation by analyzing a learned model or by constructing a simpler model that captures the essence of the learned model. A recent approach for VQA introduced a captioning model to generate an image's description, reasoned with the caption and the question to answer the question, and used the caption to explain the answer [11]. However, these algorithms do not support the use of domain knowledge to speed up learning or to provide intuitive explanations.

2.2 Reducing Training Data Requirements

The data required to train a deep network can be reduced by focusing on data relevant to the task at hand. One recent approach for VQA uses a Long Short-Term Memory (LSTM) network to map the question to an encoded vector, extracts a feature map from the input image using a CNN, and uses a neural network to compute weights for feature vectors from image regions based on their relevance to the question. A stacked attention network is trained to map the weighted feature vectors and question vector to the answer [25]. Researchers have also developed a co-attentional model that uses information from the question to identify relevant image regions, and uses information from the image to identify relevant words in the question [13]. In other work, active learning has been used to reduce the amount of annotations required in the dataset [12]. A model trained on an initial training set is revised iteratively by expanding the training set with image-question pairs involving concepts it is uncertain about, with an "oracle" (human annotater) providing the answers. Although this approach has been shown to reduce annotation time by $\approx 20\%$, the database needs to include just as many images and questions as in the absence of this approach.

2.3 Reasoning with Knowledge

In computer vision, robotics and other applications, learning from data can often be made more efficient by reasoning with prior knowledge about the domain. In the context of VQA, reasoning with domain knowledge (about scene objects) has been used to answer common questions about scene objects, significantly expanding the range of natural language questions that can be answered without making the training data requirements impractical [24]. However, this approach does not reduce the amount of data required to train the corresponding deep network. Another approach for VQA directly used a knowledge base to answer questions and did not consider the corresponding images as inputs [5]. One recent promising approach for VQA used physics engines and prior knowledge (of domain objects) to realistically simulate and explore different situations. These simulations guided the training of deep network models that anticipate action outcomes and answer questions about hypothetical situations [23].

In summary, deep networks are the state of the art for VQA (and many other tasks), but it is difficult to provide efficient learning and intuitive explanations with such networks. To address these problems, we use reasoning with commonsense knowledge and inductive learning to guide deep learning, as described below.

3 ARCHITECTURE

Figure 1 is an overview of our architecture that provides answers to explanatory questions about scenes and an underlying classification problem. The architecture is designed to minimize training effort (i.e., training time and number of training samples) and improve accuracy—it may be viewed as having three components.

(1) CNN-based feature extractors are trained and used to map any given image to a vector of image features.

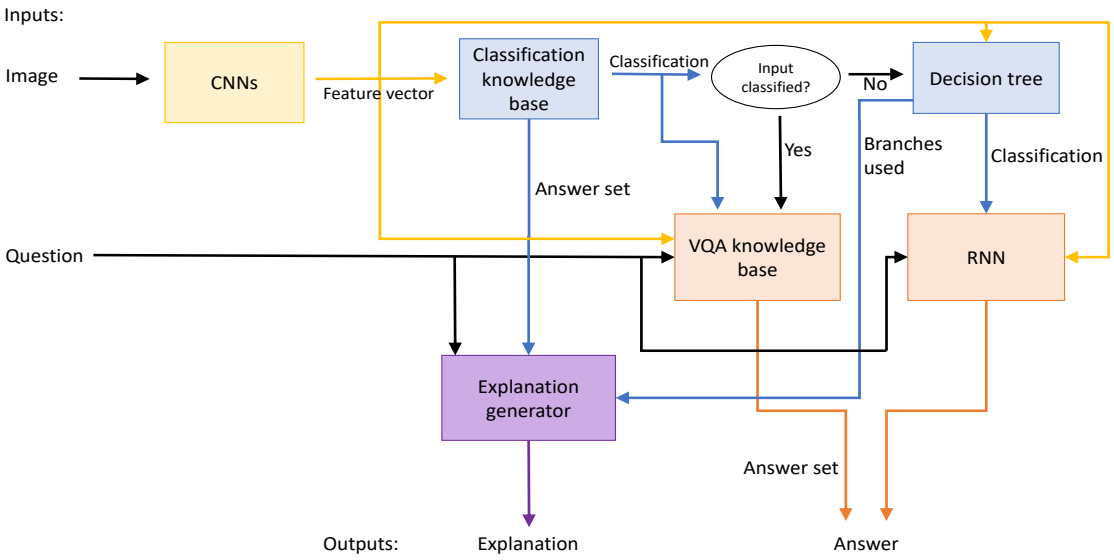

Figure 1: Overview of the components of the proposed architecture that combines the principles of deep learning, non-monotonic reasoning, and decision-tree induction.

(2) A component that chooses between two methods to classify the vector of image features. First, Answer Set Prolog (ASP) is used to reason with incomplete domain knowledge in order to assign a class label and explain this decision. If ASP-based reasoning is unable to classify the image, a decision tree classifier is trained and used to map the feature vector to a class label and explain the classification.

(3) A component to answer explanatory questions about the scene. If ASP-based reasoning is used for classification, it is also used to answer these questions. If the decision tree is used for classification, an RNN is trained to map the decision tree's output, the image features, and the question, to the corresponding answer.

This architecture exploits the complementary strengths of deep learning, non-monotonic logical reasoning, and decision tree induction. Reasoning with commonsense knowledge guides learning, e.g., the RNN is trained on (and processes) input data that cannot be processed using existing knowledge. The CNNs can be replaced by other image feature extractors. Also, although the CNNs and RNN are trained in an initial phase in this paper, these models can be revised over time if needed. The overall architecture and methodology are generic and can be applied to other domains. We hypothesize that embedding logical reasoning and the decision tree classifier between the CNNs and RNN makes the decisions more interpretable, and makes learning more time and sample efficient. We illustrate and evaluate the architecture's components and the methodology using the following two running examples:

(1) **Structure Stability (SS):** this domain has different structures, i.e., different arrangements of simulated blocks of different colors and sizes on a tabletop, e.g., see Figure 2. We generated 2500 such images using a physics-based simulator. The relevant features of the domain include the number of blocks, whether the structure is on a lean, whether the structure has a narrow base, and whether any block is displaced (placed so far to the side) such that it is not well balanced on top of the block below. The objective is to classify structures as being stable or unstable, and to answer explanatory questions such as "why is this structure unstable?" and "what needs to happen to make the structure stable?".

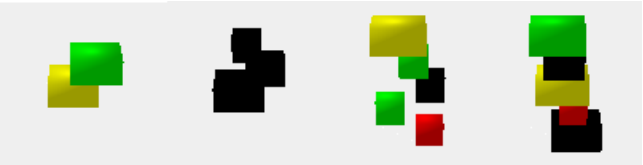

Figure 2: Illustrative images of structures of blocks of different colors and sizes from physics-based simulator.

Figure 3: Illustrative images of traffic signs from the BelgiumTS dataset [22].

(2) **Traffic Sign (TS):** this domain focuses on recognizing traffic signs from images—see Figure 3. We used the BelgiumTS benchmark dataset [22] with ≈ 7000 real-world images (total) of 62 different traffic signs. This domain's features include the primary symbol/sign in the middle, secondary symbol, shape (e.g., circle, hexagon), color (main, border), background

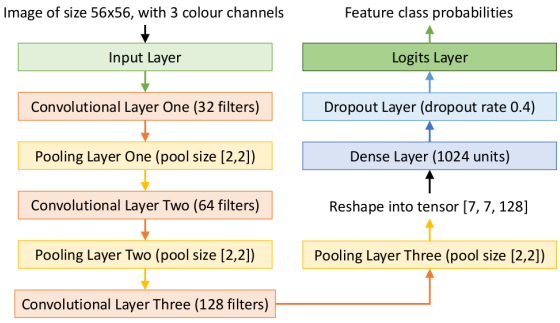

Figure 4: Illustrative example of the CNN architecture used for feature extraction in our architecture. CNNs for individual features may have have different numbers of convolutional and pooling layers.

image etc. The objective is to classify the traffic signs and to answer explanatory questions such as "what is the sign's message?" and "how should the driver respond to this sign?".

3.1 CNN for Feature Extraction

The first component of the architecture maps input images to concise features representing the objects of interest. For our example domains (SS and TS), semi-automated annotation was used to obtain the relevant features from images of different scenes. The features for each domain were selected based on domain expertise.

To minimize the need for training data, and to simplify the training of CNNs, we (i) train a separate CNN for each of the desired features to be extracted from an image; and (ii) start with a basic CNN architecture and incrementally make it more complex as needed. The number of CNNs is thus equal to the number of features to be extracted from each image, and the structure of the CNN trained for each feature may be different. The basic CNN architecture has an input layer, a convolutional layer, a pooling layer, a dense layer, a dropout layer, and a logit layer. Additional convolutional and pooling layers are added until the feature extraction accuracy converges (or exceeds a threshold). For more complex features, we also explore fine-tuning previously trained CNN models instead of starting from scratch. One CNN architecture learned for feature extraction in our example domains has three convolutional layers and pooling layers, as shown in Figure 4. In this work, these CNNs were trained in an initial phase and then used for testing.

3.2 Classification with ASP or Decision Tree

Once the feature vector has been extracted from an image, it needs to be assigned a domain-dependent class label. In our architecture, we assign this class label using non-monotonic logical reasoning or a decision tree-based classifier.

ASP-based Inference: ASP, a declarative programming paradigm, is used to represent and reason with incomplete commonsense domain knowledge. ASP is based on stable model semantics, and supports *default negation* and *epistemic disjunction*, e.g., unlike "$\neg a$" that states *a is believed to be false*, "*not a*" only implies *a is not believed to be true*, and unlike "$p \vee \neg p$" in propositional logic, "*p or* $\neg p$" is not tautological. Each literal can thus be true, false or unknown, i.e., the agent does not have to believe anything that it is not forced to believe. ASP can represent recursive definitions, defaults, causal relations, special forms of self-reference, and language constructs that occur frequently in non-mathematical domains, and are difficult to express in classical logic formalisms [6]. Also, unlike classical first-order logic, ASP supports non-monotonic logical reasoning, i.e., it can revise previously held conclusions (or equivalently reduce the set of inferred consequences) based on new evidence, aiding in the recovery from errors due to the incomplete knowledge. Approaches that reason with domain knowledge are often criticized for requiring considerable prior knowledge, and for being unwieldy in large, complex domains. However, modern ASP solvers support efficient reasoning in large knowledge bases, and are used by an international research community for cognitive robotics [4, 26] and other applications [3]. Furthermore, existing work can be used to incrementally and interactively learn (or revise) such symbolic domain knowledge [20].

An ASP *program* (Π) has a *sorted signature* Σ and axioms. The signature Σ includes *sorts*, *statics*, i.e., domain attributes that do not change over time, and *fluents*, i.e., domain attributes whose values can be changed. For instance, in the SS domain, Σ includes sorts such as *block*, *color*, and *size*, and sorts of the TS domain include *sign*, *main_color*, *other_color*, *main_symbol*, *other_symbol* etc. Σ also has the sort *step* for temporal reasoning. Statics and fluents in our example domains (SS, TS) include:

$$\begin{aligned} &num\_blocks(struc, num),\ block\_color(block, color), \\ &block\_size(block, size) \\ &primary\_symbol(sign, main\_symbol) \\ &primary\_color(sign, main\_color) \\ &secondary\_symbol(sign, other\_symbol) \end{aligned} \tag{1}$$

These relations are described in terms of their arguments' sorts. In addition, predicate $holds(fluent, step)$ implies that a particular fluent holds true at a particular timestep.

The axioms of Π encode rules governing domain dynamics. Some axioms in our example domains (SS, TS) include:

$$unstable(S) \leftarrow block\_displaced(S) \tag{2a}$$

$$\begin{aligned} stable(S) \leftarrow\ & num\_blocks(S, 2), \\ & \neg struc\_type(S, lean) \end{aligned} \tag{2b}$$

$$\begin{aligned} sign\_type(TS, no\_parking) \leftarrow\ & primary\_color(TS, blue), \\ & primary\_symbol(TS, blank) \end{aligned} \tag{2c}$$

where Statement 2(a) says that any structure with a block that is displaced significantly is unstable, and Statement 2(b) says that any pair of blocks that does not have a significant lean is stable. Statement 2(c) says that a traffic sign that is blue and blank is a no parking sign. We also encode defaults that hold in all but a few exceptional circumstances, e.g., "structures with two blocks of the same size are usually stable".

The accuracy of the inferences drawn from the encoded knowledge depends on the accuracy and extent of knowledge encoded in the ASP program, but encoding comprehensive domain knowledge is difficult. The decision of what (and how much) knowledge to encode is made by the designer (i.e., domain expert). Also, in dynamic domains, Π includes *actions* with their preconditions and effects; a

history of observations and executed actions is also considered. We leave the exploration of these capabilities for future work and do not describe them here; see [6] for more details.

Once the features extracted from an input image are encoded as the initial state of the domain, the ground literals in an *answer set* obtained by solving Π represent the beliefs of an agent associated with Π. All reasoning (e.g., planning) can be reduced to computing answer sets of Π [6]. We use the SPARC system [2] to compute answer set(s) of ASP programs. The relevant literals in the answer set provide the class label and an explanation for this choice.

Decision Tree Classifier: If ASP-based inference cannot classify the feature vector extracted from an image, it is classified using a decision tree classifier learned from such training examples. We use a standard implementation of a decision tree that uses the Gini measure to compute information gain and identify the features to split on at each level of the tree. Since the decision tree's search space is quite specific (samples not classified by ASP-based reasoning), it does not need to generalize as much as it would have to if it had to process every training (or test) sample in the dataset. Also, although overfitting is much less likely, it is still possible; pruning can then be used to minimize the effects of overfitting. Figure 5 shows part of a learned decision tree classifier; specific nodes used to classify a particular example are highlighted. These "active" nodes can be used to provide an explanation for the class label assigned to a specific image under consideration.

3.3 Answering Explanatory Questions

The third component of the architecture provides two methods for answering explanatory questions. The available inputs are the transcribed question, vector of image features and the classification output. The human designer provides use pre-determined templates for questions and their answers, e.g., we use fixed keywords and a controlled vocabulary. Any given question is transcribed, parsed and matched with these templates to obtain a relational representation. Examples of questions in the SS domain include: "is this structure stable/unstable?", " what is making this structure stable/unstable?", and "what would need to be changed to make this structure stable/unstable". Examples of questions in the TS domain include: "what sign is this?", "what is the sign's message?", and "how should the driver respond to this sign?".

The first method for answering explanatory questions is based on the understanding that if the feature vector (extracted from the image under consideration) is classified using ASP-based reasoning, then there is sufficient knowledge in the ASP knowledge base to answer explanatory questions about the scene. To support such question answering, we augment the signature Σ and axioms in system description $\mathcal{D}$. For instance, we add sorts such as *query_type*, *answer_type*, and *query*, and suitable relations to represent questions, answers, and and more abstract attributes (e.g., of structures of blocks, traffic signs etc). We also include axioms such as:

$$many\_blocks(S) \leftarrow unstable(S),\ \neg base(S, narrow),\ \neg struc\_type(S, lean),\ block\_displaced(S) \quad (3)$$

which implies that if a structure (of blocks) is not on a narrow base, does not have a significant lean, and does not have blocks significantly displaced, then any instability is because there are too many blocks stacked together. We also ensure that a question does not result in multiple answers that contradict each other. Once the program Π has been augmented, we can (as before) compute answer set(s) of Π. For any given question, the answer set(s) are parsed to extract literals that are used to construct the answers based on the pre-determined templates.

The second method for providing answers to explanatory questions is used if the decision tree is used to classify the vector of image features. ASP-based reasoning's inability to classify the feature vector is taken to imply that there is insufficient knowledge in the ASP program to answer explanatory questions about the scene. In this case, an LSTM network-based RNN is trained and used to answer explanatory questions. This RNN takes as input the feature vector, classification output, and a vector representing the transcribed and parsed query. The output (provided during training) is in the form of answers in the predetermined templates. As with the CNN for feature extraction, the network architecture is built incrementally during training. We begin with a single hidden layer and increase complexity until the accuracy exceeds a threshold. We also provide the option of adding a stack of LSTMs if adding individual layers does not improve network accuracy significantly. In our example domains, the RNN constructed to answer explanatory questions had as many as 24 hidden layers and uses a softmax function to provide one of about 50 potential answer types.

4 EXPERIMENTAL SETUP AND RESULTS

In this section, we describe the results of experimentally evaluating the following hypotheses about the capabilities of our architecture:

(1) **H1**: the proposed architecture outperforms architectures based on just deep networks when the size of the training dataset is small;
(2) **H2**: the proposed architecture does at least as well as an architecture based on deep networks as the size of the training dataset grows; and
(3) **H3**: the proposed architecture provides intuitive answers to explanatory questions about the scene.

These hypotheses were evaluated in the context of the SS and TS domains introduced in Section 3. We provide execution traces in support of hypothesis $H3$, and include quantitative results in support of $H1$ and $H2$. For the quantitative experimental comparison, we use accuracy (precision) as the performance measure. The accuracy of the answers to explanatory questions was evaluated heuristically by computing whether the answer mentions all image attributes relevant to the classification problem. This relevance was established by a human expert, which was the lead author of this paper for the results reported below.

In the initial setup phase, we used two-thirds of the available data to train the deep networks and other computational models, using the remaining one-third for testing. For each image, we randomly chose from the set of suitable questions. We repeated this process multiple times and report the average of these trials below.

4.1 Execution Traces

The following execution traces illustrate our architecture's ability to reason with commonsense knowledge and learned models to provide intuitive answers for explanatory questions.

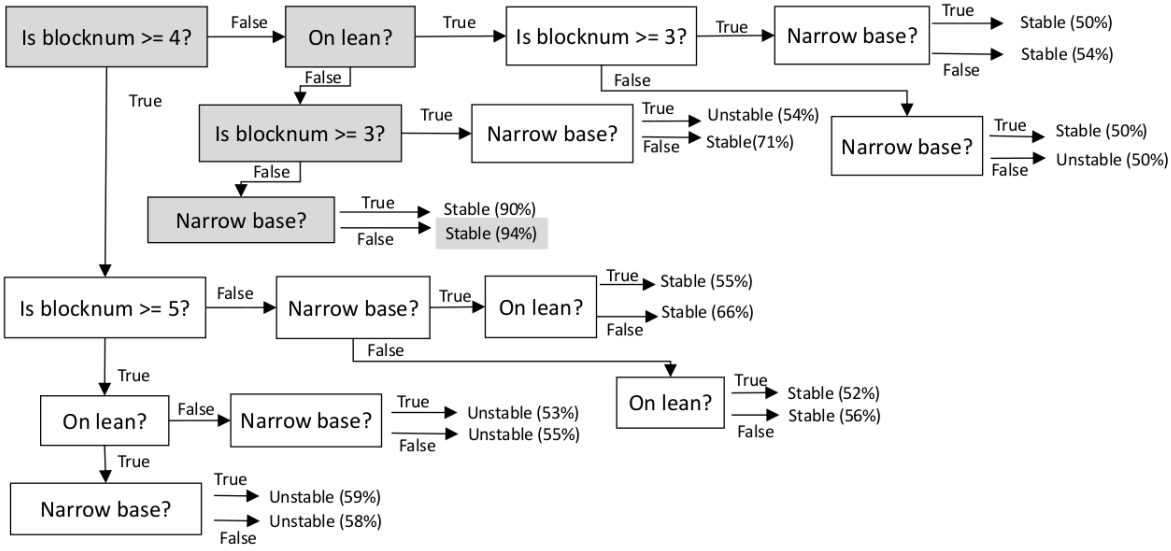

Figure 5: Part of a decision tree constructed for classification. Nodes used to classify a particular example are highlighted.

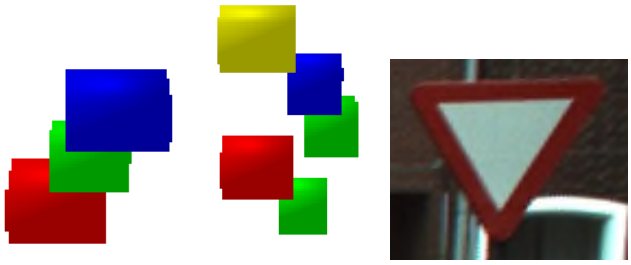

Figure 6: Images referenced in Execution Example 1 and 2.

Execution Example 1. *[Question Answering in SS domain]* Consider a scenario in the SS Domain in which the input (test) image is that shown on the left in Figure 6.

- The question posed is: "what is making this structure stable?"
- The architecture's answer: "It has a small number of blocks, a wide base, and the blocks are well balanced."
- This answer is based on the following features extracted by CNNs from the image: (i) three blocks; (ii) wide base; (iii) standing straight; and (iv) all blocks in place.
- The extracted features were converted to literals. Since ASP-based inference could not provide a conclusive answer about stability based on these literals, the example was classified using the trained decision tree (as being stable)[1]. Figure 5 shows some of the features used to obtain this class label. These features are also used to explain the classification output.
- The decision tree output, image feature vector and question, were processed by the previously trained RNN to provide the answer type and the particular answer described above.

[1] ASP-based inference was able to classify (as being stable) and answer questions about the image in the center of Figure 6.

Execution Example 2. *[Question Answering in TS domain]* Consider a scenario in the TS Domain in which the input (test) image is that shown on the right in Figure 6.

- The question posed is: "what is the sign's message?"
- The architecture's answer: "please yield to cross traffic."
- As an explanation for this answer, the architecture identifies that the CNNs extracted the following features of the sign in the image: (i) it is inverted-triangle shaped; (ii) main color is white and other (i.e., border) color is red; (iii) it has no background image; and (iv) it has a blank symbol and no secondary symbol.
- These features were converted to literals and used in ASP-based inference. In this case, ASP-based inference classifies the sign correctly (as "yield sign").
- The output literals in the corresponding answer set were inserted into a suitable template to provide the particular answer described above.

These (and other such) execution traces support hypothesis $H3$.

4.2 Experimental Results

To quantitatively evaluate hypotheses $H1$ and $H2$, we ran experimental trials in which we varied the size of the training dataset. In these trials, baseline performance was provided by a CNN-RNN architecture, with the CNNs processing images to extract and classify features, and the RNN providing answers to explanatory questions. The corresponding results are summarized in Figures 7 and 8 for the SS domain, and in Figures 9 and 10 for the TS domain. We make some observations based on these figures:

- The performance of the proposed architecture is better than an architecture based on just deep networks when the size of the training set is small, especially when we consider the ability to answer explanatory questions.

Figure 7: Classification accuracy as a function of the number of training samples in the SS domain.

Figure 8: VQA accuracy as a function of the number of training samples in the SS domain.

- The classification accuracy increases with the size of the training set[2]. This is especially true if the variability (of the features) of the domain is limited. For instance, in the relatively simpler SS domain, the CNN (on its own) performs better than our (more sophisticated) approach even with small training sets—see Figure 7.
- VQA performance, on the other hand, does not improve just by increasing the size of training set, even in simpler domains, e.g., see Figure 8. This is because VQA performance also depends on the complexity of the explanatory questions. For more complex domains, the VQA accuracy does not really increase significantly just by increasing the size of the training set, e.g., see Figure 10.

We explored the statistical significance of the observed VQA performance by running paired two-tailed t-tests. We observed that the VQA performance of the proposed architecture was significantly better than that of the baseline architecture; this is more pronounced in the TS domain that is more complex than the SS domain. Also, although the baseline architecture provides better classification performance, the difference is (for the most part) not statistically significant.

To further explore the observed results, we obtained a "confidence value" from the logits layer of each CNN used to extract a feature from the input image. For each CNN, the confidence value is the largest probability assigned to any of the possible values of the corresponding feature, i.e., it is the probability assigned to the most likely value of the feature. These confidence values are considered to be a measure of the network's confidence in the corresponding features being a good representation of the image. If the confidence value for any feature was low, the image features were only used to revise the decision tree (during training), or were processed using the decision tree (during testing). We hypothesized that this approach would improve the accuracy of classification and question answering, but it did not make any significant difference in our experimental trials. Furthermore, although we do not discuss it here, we used our architecture on a physical robot collaborating with a

[2]We also ran experiments with larger datasets, which we do not report in this paper.

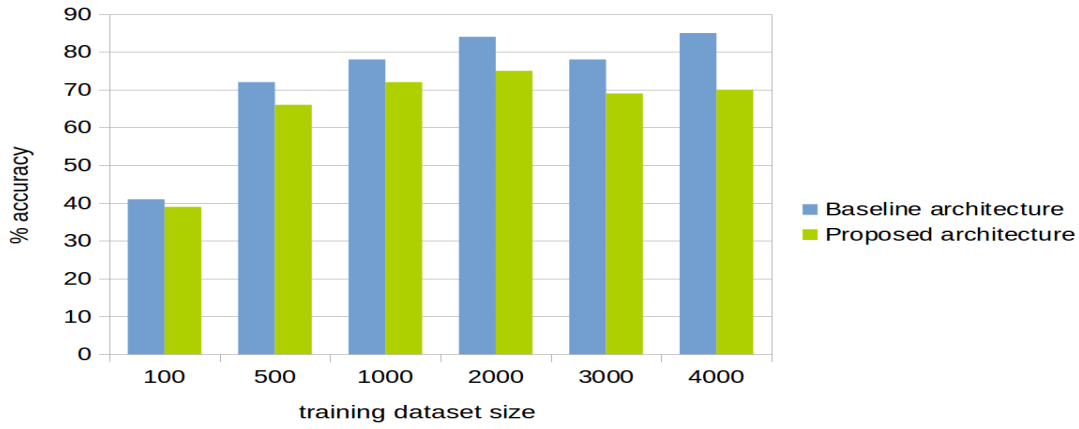

Figure 9: Classification accuracy as a function of number of training samples in TS domain.

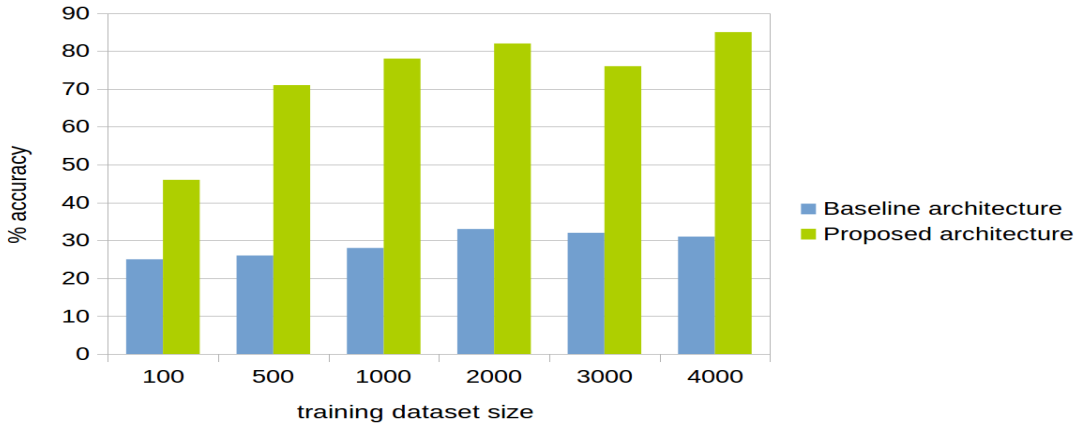

Figure 10: VQA accuracy as a function of number of training samples in TS domain.

human to jointly manipulate and describe structures of blocks on a table, i.e., in scenarios similar to the SS domain.

5 CONCLUSIONS

Visual question answering (VQA) combines challenges in computer vision, natural language processing, and explainability. In addition to improving trust in the use of machine learning algorithms in critical application domains, explainability helps design better algorithms. State of the art algorithms for VQA are based on deep learning; these are computationally expensive, require large training datasets, and make it difficult to support explainability. Inspired by research in cognitive systems, the architecture described in this paper exploits the complementary strengths of deep learning, non-monotonic logical reasoning with commonsense knowledge, and decision tree induction. Experimental results indicate that the proposed architecture outperforms an architecture based on just deep networks for smaller training datasets, provides comparable performance as the dataset grows larger, and provides intuitive answers to explanatory questions.

The architecture opens up multiple directions of future work. First, we will explore the use of our architecture in other domains with datasets of increasing size and complexity (in terms of the features and the questions). Second, we will explore different deep network structures in our architecture, using the explanatory answers to further understand the internal representation of these networks. Furthermore, we are interested in combining the VQA architecture with our cognitive architectures for knowledge representation, reasoning and learning [19, 20], thus enabling a robot to represent, reason, and learn while collaborating with humans in complex domains.

ACKNOWLEDGMENTS

The authors thank Ales Leonardis for feedback on the architecture described in this paper. This work was supported in part by the US Office of Naval Research Science of Autonomy award N00014-17-1-2434, and the Asian Office of Aerospace Research and Development award FA2386-16-1-4071.

REFERENCES

[1] A. Agrawal, D. Batra, D. Parikh, and A. Kembhavi. 2017. Don't Just Assume; Look and Answer: Overcoming Priors for Visual Question Answering. In *International Conference on Computer Vision and Pattern Recognition*. Honolulu, USA.

[2] Evgenii Balai, Michael Gelfond, and Yuanlin Zhang. 2013. Towards Answer Set Programming with Sorts. In *International Conference on Logic Programming and Nonmonotonic Reasoning*. Corunna, Spain.

[3] Esra Erdem, Michael Gelfond, and Nicola Leone. 2016. Applications of Answer Set Programming. *AI Magazine* 37, 3 (2016), 53–68.

[4] Esra Erdem and Volkan Patoglu. 2012. Applications of Action Languages to Cognitive Robotics. In *Correct Reasoning*. Springer-Verlag.

[5] Ulrich Furbach, Ingo Glöckner, Hermann Helbig, and Björn Pelzer. 2010. Logic-Based Question Answering. *KI - Künstliche Intelligenz* 24 (2010), 51–55.

[6] Michael Gelfond and Yulia Kahl. 2014. *Knowledge Representation, Reasoning and the Design of Intelligent Agents*. Cambridge University Press.

[7] Y. Goyal, T. Khot, D. S. Stay, D. Batra, and D. Parikh. 2017. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. In *International Conference on Computer Vision and Pattern Recognition*. Honolulu, USA, 6325–6334.

[8] A. Jabri, A. Joulin, and L. van der Maaten. 2016. Revisiting Visual Question Answering Baselines. In *European Conference on Computer Vision*. Amsterdam.

[9] Aiwen Jiang, Fang Wang, Fatih Porikli, and Yi Li. 2015. *Compositional Memory for Visual Question Answering*. Technical Report. Available at: https://arxiv.org/abs/1511.05676.

[10] Pang Wei Koh and Percy Liang. 2017. Understanding Black-box Predictions via Influence Functions. In *International Conference on Machine Learning (ICML)*. Sydney, Australia, 1885–1894.

[11] Qing Li, Jianlong Fu, Dongfei Yu, Tao Mei, and Jiebo Luo. 2018. *Tell-and-Answer: Towards Explainable Visual Question Answering using Attributes and Captions*. Technical Report. Available at: https://arxiv.org/abs/1801.09041.

[12] T. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. Zitnick, and P. Dollar. 2014. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*.

[13] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical Question-Image Co-Attention for Visual Question Answering. In *Advances in Neural Information Processing Systems*. Barcelona, Spain.

[14] Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. 2017. Ask Your Neurons: A Deep Learning Approach to Visual Question Answering. *International Journal of Computer Vision* 125 (2017), 110–135. Issue 1-3.

[15] Issey Masuda, Santiago Pascual de la Puente, and Xavier Giro i Nieto. 2016. Open-Ended Visual Question-Answering. In *International Conference on Computer Vision and Pattern Recognition*. Las Vegas, USA.

[16] Supriya Pandhre and Shagun Sodhani. 2017. *Survey of Recent Advances in Visual Question Answering*. Technical Report. Available at: https://arxiv.org/abs/1709.08203.

[17] Marco Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why Should I Trust You? Explaining the Predictions of Any Classifier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. San Francisco, USA, 1135–1144.

[18] R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. In *International Conference on Computer Vision*. Venice, Italy, 618–626.

[19] Mohan Sridharan, Michael Gelfond, Shiqi Zhang, and Jeremy Wyatt. 2018. *REBA: A Refinement-Based Architecture for Knowledge Representation and Reasoning in Robotics*. Technical Report. Availble at: http://arxiv.org/abs/1508.03891.

[20] Mohan Sridharan and Ben Meadows. 2018. Knowledge Representation and Interactive Learning of Domain Knowledge for Human-Robot Collaboration. In *International Conference on Advances in Cognitive Systems*. Stanford, USA.

[21] Damien Teney and Anton van den Hengel. 2016. *Zero-Shot Visual Question Answering*. Technical Report. Available at: https://arxiv.org/abs/1611.05546.

[22] Radu Timofte, Markus Mathias, Rodrigo Benenson, and Luc Van Gool. 2013. Traffic Sign Recognition - How far are we from the Solution?. In *International Joint Conference on Neural Networks (IJCNN)*. Dallas, USA, 1–8.

[23] Misha Wagner, Hector Basevi, Rakshith Shetty, Wenbin Li, Mateusz Malinowski, Mario Fritz, and Ales Leonardis. 2018. Answering Visual *What-If* Questions: From Actions to Predicted Scene Descriptions. In *Visual Learning and Embodied Agents in Simulation Environments (VLEASE) Workshop at ECCV*. Munich, Germany. Available on arXiv: https://arxiv.org/abs/1809.03707.

[24] Peng Wang, Qi Wu, Chunhua Shen, Anton van den Hengel, and Anthony R. Dick. 2017. Explicit Knowledge-based Reasoning for Visual Question Answering. In *International Joint Conference on Artificial Intelligence*. Melbourne, Australia.

[25] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alexander J. Smola. 2016. Stacked Attention Networks for Image Question Answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, USA, 21–29.

[26] Shiqi Zhang, Mohan Sridharan, and Jeremy Wyatt. 2015. Mixed Logical Inference and Probabilistic Planning for Robots in Unreliable Worlds. *IEEE Transactions on Robotics* 31, 3 (2015), 699–713.

[27] Ted Zhang, Dengxin Dai, Tinne Tuytelaars, Marie-Francine Moens, and Luc Van Gool. 2017. *Speech-Based Visual Question Answering*. Technical Report. Available at: https://arxiv.org/abs/1705.00464.