Learning by Asking Commonsense Questions

Yuan Yang [1] Hanlin Zhang [2]

Abstract

Modern supervised models require a large amount of labeled data to be successful. In the meta-learning context, the amount of human supervision can be reduced if the model can transfer past knowledge to facilitate future training.

In this work, we approach this challenge through a novel active learning framework that can quickly learn a new task by asking commonsense questions to the human oracle. The questions are generated as first-order logic rules that capture the common patterns of the new class with the existing concepts learned in previous tasks. The rules are highly interpretable such that human oracle can evaluate them without going through the individual samples they apply to, leading to less human supervision. Our method is evaluated on the Visual Genome dataset, which achieves the same performance with significantly fewer oracle queries than several strong baselines.

1. Introduction

One of the hallmarks of human intelligence is the ability to learn continuously, and accumulate knowledge from a series of tasks that enable future learning (Chen & Liu, 2016; Mitchell et al., 2018). Compared to the modern machine learning (ML) methods, such capability enables humans to acquire new knowledge with only a few samples. For many real-world applications, the cost of labeling a large amount of data becomes prohibitive. This raises a challenging question: *can the ML agent learn to utilize prior experiences to address a new tasks where labels are scarce?*

A wealth body of researches is proposed for advancing the current machine learning methods towards this goal. For example, meta-learning (Vinyals et al., 2016; Munkhdalai & Yu, 2017; Santoro et al., 2016; Finn et al., 2017) seeks to train a model on a variety of tasks such that the model can adapt to a new task with a few labels. Active learning (AL) methods (Joshi et al., 2009; Contardo et al., 2017; Fang et al., 2017; Bachman et al., 2017; Konyushkova et al., 2017), on the other hand, approach this by building-in the awareness of supervision cost into the model such that the cost is reduced by choosing more effective samples. Life-long learning (Mitchell et al., 2018) investigates the learning of a sequence of new tasks, where the model can transfer knowledge from previous tasks to aid the future task.

Inspired by the recent work on *learning-by-asking-questions* (Yang et al., 2018; Shen et al., 2019; Misra et al., 2018).We consider this problem in an interactive learning scenario, which we refer to as the *learning-by-asking-commonsense-question* framework. In this scenario, our agent is tasked to learn a set of new concepts by asking questions to the human oracles. And similar to the meta-learning, the agent can access a knowledge graph which encodes the existing knowledge on the related concepts and relations. The goal is to utilize the knowledge graph such that the generated questions lead to a more efficient learning on the new tasks.

To achieve this goal, our agent is capable of discovering the common patterns of the new concept by associating it with the existing ones on a knowledge graph. Such patterns are represented as first-order logic rules which guarantee to generalize to many samples. To interact with the human oracle, the agent asks the commonsense questions that are translated from the rules. Evaluating such question is efficient, since the human oracle does not need to inspect the individual instances that the rule applies to, which leads to a reduction in human supervision.

We evaluate our method on a set of visual object classification tasks together with several strong AL and meta-learning baselines. Our method can generalize to new classes with significantly fewer queries to the oracle.

2. Preliminaries

2.1. Knowledge representation

In this work, we consider the knowledge and data as a set of object instances and relations, essentially in the form of a knowledge graph (KG). For example, Visual Genome (Krishna et al., 2016) consists of 100K images over 80K object

[1]Georgia Institute of Technology [2]South China University of Technology. Correspondence to: Yuan Yang <yyang754@gatech.edu>.

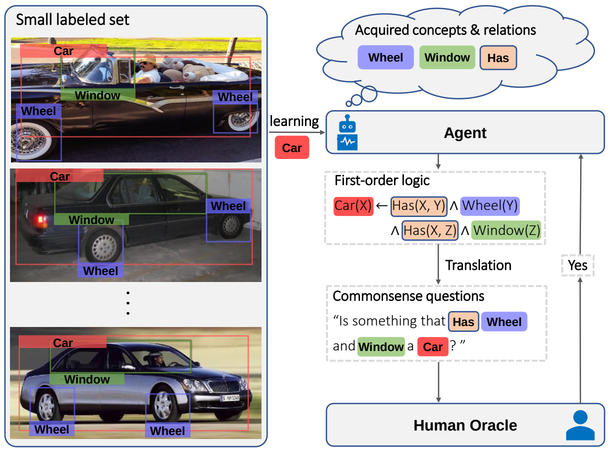

Figure 1: Learning a new class Car by asking commonsense questions to the human oracle. For each new task, the agent is provided with a small set of labeled target class. The agent learns to capture the patterns of the new concept with those concepts and relations that it already acquired, and represent them into first-order logic rules. The rules are translated into commonsense questions and sent to the oracle for labeling.

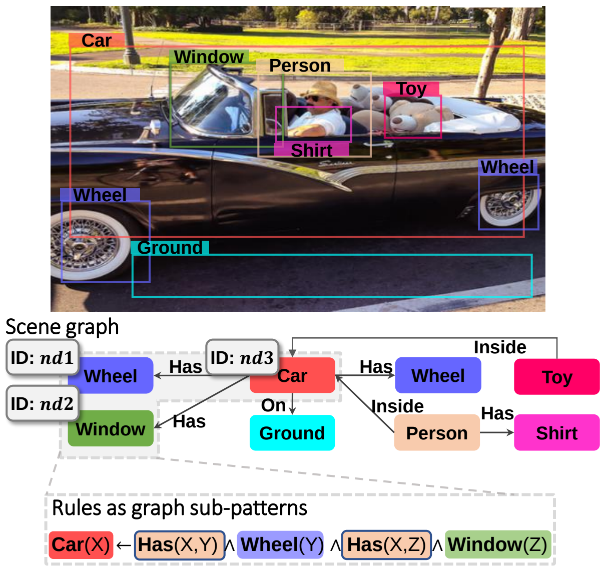

Figure 2: In Visual Genome dataset, images are annotated with objects and the relations among them. This information can be represented as a scene-graph. And the goal of our method is to recognize the sub-patterns (i.e. rules) in the graph with respect to the target class Car.

classes. As shown in Figure 2, apart from the pixel information, each image in the dataset is also described with a *scene-graph*. The graph consists of objects as the graph nodes and their spatial or semantic relations as graph edges.

Formally, a KG is defined over a set of predicates $P \in \mathcal{P} = \mathcal{U} \cup \mathcal{B}$ and entity space $\boldsymbol{x} \in \mathcal{X}$ where $\boldsymbol{x} \in \mathbb{R}^d$. Object classes are unary predicates $P \in \mathcal{U}$ such that each P is a mapping $P : \mathbb{R}^d \mapsto s \in [0, 1]$, and relations are binary predicates $P \in \mathcal{B}$ with mapping $P : \mathbb{R}^d \times \mathbb{R}^d \mapsto s \in [0, 1]$. By using the one-hot entity encoding $\boldsymbol{v_x} \in \{0, 1\}^{|\mathcal{X}|}$, each predicate P can be parameterized into an adjacency matrix $\boldsymbol{M} \in \{0, 1\}^{|\mathcal{X}|\times|\mathcal{X}|}$, where $m_{ij} = 1$ indicates $\langle \boldsymbol{x}_i, P, \boldsymbol{x}_j \rangle$ exists in the graph. In the case of unary predicate, the matrix is diagonal such that $m_{ii} = 1$ indicates $\langle \boldsymbol{x}_i, P, \boldsymbol{x}_i \rangle$ (we duplicate the same variable for notation consistency) exists in the KG.

2.2. First-order logic rules as graph sub-patterns

Given the KG, one can express knowledge as first-order logic (FOL) rules. A FOL rule consists of (i) a set of predicates defined in $\mathcal{P}$, (ii) a set of logical variables such as X, Y and Z, and (iii) logical operations $\{\wedge, \vee, \neg\}$. For example,

$$\begin{aligned} \texttt{Car}(X) \leftarrow & \texttt{Has}(X, Y) \wedge \texttt{Wheel}(Y) \\ & \wedge \texttt{Has}(X, Z) \wedge \texttt{Window}(Z) \end{aligned} \quad (1)$$

involves predicates Car, Has, Wheel and Window. Components such as $\texttt{Car}(X)$ are called **atoms** which correspond to the predicates that apply to the logical variables. Each atom can be seen as a lambda function with its logical variables as input. This function can be evaluated by **instantiating** the logical variables such as X into the object in $\mathcal{X}$. For example, in Figure 2, we can evaluate $\texttt{Car}(nd3/X)$ by instantiating X into node $nd3$. This yields 1 because the node $nd3$ is labeled as a Car. Similarly, we have $\texttt{Car}(nd1/X) = 0$ and for binary case $\texttt{Has}(nd3/X, nd1/Y) = 1$.

The outputs of each atoms are combined using logical operations $\{\wedge, \vee, \neg\}$ and the *imply* operation $p \leftarrow q$ is equivalent to $p \vee \neg q$. Thus, when all variables are instantiated, the rule will produce an output as the specified combinations of those from the atoms. For example, let $r(X, Y, Z)$ denotes Eq.(1), then we have $r(nd3/X, nd1/Y, nd2/Z) = 1$. In this way, the rule is essentially encoded to represent the knowledge that "a car is something that has wheels and windows". And through the usage of logical variables, the statement becomes a *commonsense* as it represents the "lifted" knowledge that does not depend on the specific data (Yang & Song, 2020).

Such representation is beneficial because (i) the rules are highly interpretable and can be translated into natural language for human assessment, and (ii) the rules guarantee to generalize to many examples.

3. Problem Set-up

We imagine the agent learns similarly as in human learning. In the beginning, the agent has access to a background

KG which consists of the concepts (e.g. `Wheel`, `Window`) and relations it already knows. When presented with a new concept (e.g. `Car`), the agent tries to associate it with the known concepts and generalize by proposing the commonsense statement that reflects the common pattern, e.g. "Is something that has `wheels` and `windows` a `Car`?". From the teacher's perspective, answering such a statement is more effective than showing the individual `Car` images to the learner, as the former guarantees to generalize to all `Car` samples whenever `wheel` and `window` are present.

In summary, the goal of our method lies at the intersection of lifelong learning and active learning: **that is to utilize the background knowledge to facilitate the learning of a new task such that the human supervision is minimal**.

Formally, the agent is given a knowledge graph $\mathcal{KG} = \langle \mathcal{P}_{\mathcal{KG}}, \mathcal{D}_{\mathcal{KG}} \rangle$ that consists of facts $\mathcal{D}_{\mathcal{KG}} = \{\langle \boldsymbol{x}, P, \boldsymbol{x}' \rangle\}$ for predicates $P \in \mathcal{P}_{\mathcal{KG}}$. We define the learning problem as learning a set of new tasks $L = \{L_1, .., L_n\}$. Each task is defined as $L_i = \langle P_i, \mathcal{D}_i, \mathcal{H}_i, Y_i \rangle$, where P_i is the target predicate to be learned, $\mathcal{D}_i$ is a small set of labeled data for warming-up, and the agent has the access to the rest of unlabelled data in $\mathcal{H}_i$ whose ground-truth labels are stored in Y_i. Here, we assume the target predicates are all unary, i.e. each task is a visual object classification problem. But this can readily generalize to binary cases as well, which corresponds to a relational learning problem.

The agent learns the target object classes by training a *learner model* (e.g. MLP) $f_{\boldsymbol{w}}(\boldsymbol{x}) : \mathcal{X} \mapsto s \in [0, 1]$ parameterized by $\boldsymbol{w}$[1]. Apart from learning from the labelled data in $\mathcal{D}_i$, the agent can also generate queries $q \in Q_i$ with a policy model $\pi_\theta(q|\mathcal{H}_i, \mathcal{KG})$ that is parameterized by θ. Queries are submitted to a human oracle O for evaluation and the feedback is reflected as the label information on $\mathcal{H}_i$, which we denote as $\hat{Y}_i$.

Overall, this lifelong active learning $\langle \mathcal{KG}, L, O \rangle$ aims at solving the tasks L by using the existing knowledge as well as the online supervisions from oracle O. Let $f_{\boldsymbol{w}|\hat{Y}}$ denote the learner model trained on the obtained labels from all tasks. We define the loss of each task L_i as

$$\mathcal{L}(f_{\boldsymbol{w}|\hat{Y}}, \mathcal{H}_i) = \frac{1}{|\mathcal{H}_i|} \sum_{\boldsymbol{x} \in \mathcal{H}_i, y \in Y_i} \text{XEnt}(f_{\boldsymbol{w}|\hat{Y}}(\boldsymbol{x}), y), \quad (2)$$

where y is the true label. The goal is to minimize the averaged loss of L within a fixed number of queries, i.e. *budget*.

$$\arg\min_{\boldsymbol{w}, \theta} \frac{1}{n} \sum_i \mathcal{L}(f_{\boldsymbol{w}|\hat{Y}}, \mathcal{H}_i) \quad (3)$$
$$\text{s.t.} \quad |Q_i| \leq B, \qquad i = 1, ..., n,$$

[1]Since each task involves only one class, the agent only learns a single model that does multi-class classification for all the tasks. This is different from some meta-learning works which assume different models will be learned for different tasks.

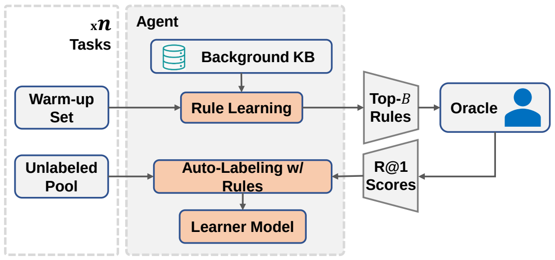

Figure 3: The procedures and components involved in the learning-by-asking-commonsense-question framework.

where B is the query budget.

3.1. Commonsense questions as queries

In traditional active learning methods, query q corresponds to a specific sample in $\mathcal{H}_i$ that needs to be labeled. This is done by treating query policy as generating a distribution over the hidden samples $\pi(\boldsymbol{x}|\mathcal{H}_i), \boldsymbol{x} \in \mathcal{H}_i$, where $\mathcal{KG}$ is not used. Alternatively, we propose to use commonsense questions as queries. Formally, apart from the learner model $f_{\boldsymbol{w}}$, the agent now maintains the query model as a rule learner $\varphi_\theta(r|\boldsymbol{x}, P_i, \mathcal{KG}), \boldsymbol{x} \in \mathcal{D}_i$. This is a parameterized model that learns on the warm-up set $\mathcal{D}_i$ and generates rules for target predicate P_i.

For each task L_i, the agent solves the learning problem with the following steps (shown in Figure 3):

- **Rule learning**: the agent fits the rule learner model φ_θ on dataset $\mathcal{D}_i$ and produces a set of rules $\mathcal{R}_i = \{r_1, r_2, ...\}$.
- **Rule evaluation**: the agent selects the top B rules $\mathcal{R}_B$ with policy $r \sim \pi(r|\mathcal{R}_i)$. The rules are translated into commonsense questions and sent to the oracle $O(r)$ for evaluation.
- **Auto-labeling with rules**: the agent uses the evaluated rules to label the data $\boldsymbol{x} \in \mathcal{H}_i$, i.e. $l(y|\boldsymbol{x}, \mathcal{R}_B, O)$, leading to a set of estimated labels $\hat{Y}_i$.
- **Training the learner model**: the agent trains the learner model $f_{\boldsymbol{w}}$ using the auto-labeled data.
- **Evaluation**: after training on all the tasks, the learner model is evaluated with Eq.(3).

4. Related Work

The learning-by-asking-commonsense-question method lies at the intersection of several domains. Here, we compare our problem set-up with those in the following domains to better illustrate our motivation.

Active learning (AL) investigates the learning problem where the samples need to be selected from an unlabeled dataset to be labeled by an oracle. Formally, for each task L_i, AL methods propose a fixed policy $\pi(q|\mathcal{H}_i)$ that guides how a sample is picked from $\mathcal{H}_i$ to learn $f_{\boldsymbol{w}}(\boldsymbol{x})$ more effectively. Common strategies include maximum entropy (Joshi et al., 2009) and expected model change (Settles et al., 2008).

Meta active learning or *learning-to-active-learn* considers the query policy generated by a parametric model and can be learned online in the active learning process (Contardo et al., 2017; Fang et al., 2017; Bachman et al., 2017; Konyushkova et al., 2017). For example, Learning Active Learning (LAL) (Konyushkova et al., 2017) proposes to learn a policy model $\pi_\theta(q|\mathcal{H}_i, \mathcal{D}_i)$. The policy is parameterized by a learnable random forest model. It selects the next sample to label by monitoring the performance of the current learner model on the revealed set and the goal is to train the policy model such that the next query improves the learner model.

Meta-learning and few-shot learning involve learning a meta-trainer that can train the learner model to quickly adapt to a new task with few labels provided (Vinyals et al., 2016; Munkhdalai & Yu, 2017; Santoro et al., 2016; Finn et al., 2017). These works are studied in the *N*-way *K*-shot learning problem which is similar to the one stated in section 3: the learner is trained on a set of background classes (i.e. $\mathcal{P}_{\mathcal{KG}}$ in our setting), and is then evaluated on *N* unseen classes where the model is allowed to fine-tune itself with *K* samples for each class. Different from AL methods, meta-learning makes use of the background data but does not select the *K*-shot samples actively, which is equivalent to a random policy π. For example, Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) learns a global parameter initialization $\bar{\boldsymbol{w}}$ on the background set, such that the learner model $f_{\boldsymbol{w}}(\boldsymbol{x}|\bar{\boldsymbol{w}})$ can generalize on the new classes with *K*-shot samples.

Learning by asking questions. Our method is related to the recent advances in *learning-by-asking-questions* (Yang et al., 2018; Misra et al., 2018). Similar to our setting, the agent in this framework learns by asking questions to the oracle. However the goal of these methods is to enable the model to interact with an open-domain environment with natural language, thus (i) the questions are localized to each data sample and (ii) the framework does not limit the number of questions can be asked by the agent. This is very different from our setting where the goal is to reduce human supervision by asking questions that generalize to many samples. (Shen et al., 2019) claims to reduce the labeling efforts but the task is restricted to the image captioning.

Lifelong learning investigates the learning of a new task with background knowledge. Many works focus on alleviating the *catastrophic forgetting* problems (Li & Hoiem, 2017; Kirkpatrick et al., 2017; Zenke et al., 2017; Lopez-Paz & Ranzato, 2017) when learning the new task. Our method is closely related to the never-ending learning (NELL) (Mitchell et al., 2018) which proposes a new learning paradigm that can continuously acquire new skills. The method represents data as knowledge graphs, and train the learner model by using the constraints posted by first-order logic (FOL) rules as the supervision signal. However, the learning tasks are pre-defined on the text data and are fixed throughout the learning process. The learning of new concepts is conducted offline, and the learned classes are integrated into the system manually.

Multi-hop reasoning on knowledge graph. Learning the common sub-patterns in the knowledge graph (i.e. the rule learner model φ_θ) is studied extensively in the literature of multi-hop reasoning (Guu et al., 2015; Lao & Cohen, 2010; Lin et al., 2015; Gardner & Mitchell, 2015; Das et al., 2016; Yang & Song, 2020). This task is generally framed as a *inductive logic programming* problem which seeks to represent these patterns into first-order logic rules. We will discuss the detailed procedure in the next section.

5. FOL Rule Learning on Scene-Graphs

We show that the rule learning on scene-graph can be formulated as *multi-hop reasoning* problem which can be solved by many existing approaches.

For the scene-graph dataset, $\mathcal{KG}$ essentially consists of a set of isolated sub-graphs each corresponding to the scene of a specific image. Let $\mathcal{X}_g$ be the entity domain of a single scene-graph $g \in \mathcal{KG}$, and similarly $\mathcal{P}_g = \mathcal{U}_g \cup \mathcal{B}_g$ be the predicate domain. For an object $\boldsymbol{x} \in \mathcal{X}_g$, we want to learn rule that predicts if it belongs to a target predicate P^*. This can be formulated as finding a relational path $\boldsymbol{x}' \xrightarrow{P^{(1)}} ... \xrightarrow{P^{(T)}} \boldsymbol{x}$ which starts from a particular node $\boldsymbol{x}' \in \mathcal{X}_g$ and ends at the target node $\boldsymbol{x}$, where $P^{(t)} \in \mathcal{B}_g, t = 1, ..., T$. Finding such path is equivalent to learning a chain-like entailment rules (Yang et al., 2017)

$$P^*(X) \leftarrow P^{(1)}(X, Y_1) \wedge ... \wedge P^{(T)}(Y_{n-1}, X'),$$

where the knowledge is encoded as "if the path exists, then $\boldsymbol{x}$ belongs to P^*". Note that the rule only encodes the positive case, meaning if the path does not exist, $P^*(\boldsymbol{x})$ is then undetermined rather than false because the rule fails to apply.

To find such rules, recall that $\boldsymbol{v}_{\boldsymbol{x}}$ denotes the one-hot vector of $\boldsymbol{x}$ with the dimension of $|\mathcal{X}_g|$. Then, we can represent each hop in the relational path as matrix multiplication, such that the (t)th hop of the reasoning along the path is computed as

$$\boldsymbol{v}^{(0)} = \boldsymbol{v}_{\boldsymbol{x}}, \qquad \boldsymbol{v}^{(t)} = \boldsymbol{M}^{(t)}\boldsymbol{v}^{(t-1)},$$

where $\boldsymbol{M}^{(t)}$ denotes the adjacency matrix of the predicate used in (t)th hop. One can verify that the jth element $v_j^{(t)}$

in $\boldsymbol{v}^{(t)}$ is the count of unique paths from $\boldsymbol{x}$ to $\boldsymbol{x}_j$ (Guu et al., 2015). After (T) steps of reasoning, we compute the score $P^*(\boldsymbol{x})$ being true as

$$\text{score}(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{v}_{\boldsymbol{x}}^\top \prod_{t=1}^{T} \boldsymbol{M}^{(t)} \cdot \boldsymbol{v}_{\boldsymbol{x}'}.$$

The goal is to find (i) a starting point $\boldsymbol{x}'$ and (ii) a common sequence of matrix multiplication $\boldsymbol{M}^{(t)}, t = 1, ..., T$, such that Eq.(4) is maximized for all the labeled samples $\boldsymbol{x} \in \mathcal{D}_i$. This problem can be relaxed as learning the weighted sums of all possible paths. We parameterize the soft node selection as

$$\boldsymbol{v}_{\boldsymbol{x}'} = \sum_{k \in \mathcal{U}_g} \beta_k^{(0)} \, \boldsymbol{M}_k \cdot \mathbf{1},$$

where $\beta^{(0)} = [\beta_1^{(0)}, .., \beta_{|\mathcal{U}_g|}^{(0)}]^\top$ is the attention vector that averages over the starting node, and $\mathbf{1}$ is the vector with elements as 1. Similarly we parameterize the soft path selection as

$$\text{score}(\boldsymbol{v}_{\boldsymbol{x}}, \alpha, \beta) = \sum_{t'=1}^{T} \alpha^{(t')} \left(\prod_{t=1}^{t'} \sum_{k \in \mathcal{B}_g} \beta_k^{(t)} \, \boldsymbol{M}_k \cdot \boldsymbol{v}_{\boldsymbol{x}'} \right), \tag{4}$$

where $\alpha = [\alpha^{(1)}, ..., \alpha^{(T)}]^\top$ is the path attention vector, and $\beta^{(t)} = [\beta_1^{(t)}, .., \beta_{|\mathcal{B}_g|}^{(t)}]^\top$ is the matrix attention vector at (t)-th step, and we denote all the attentions as (α, β).

These attentions can be generated differentiably by learning a parametric model. For example, NeuralLP (Yang et al., 2017) uses an RNN controller to generate the sequence of attention vectors with $\boldsymbol{v}_{\boldsymbol{x}}$ as the initial input. And NLIL (Yang & Song, 2020) generates attentions with a stacked Transformer module with predicate embeddings as the input.

6. Model Implementation

Rule Learner φ_θ: We use the Neural Logic Inductive Learning (NLIL) (Yang & Song, 2020) as the rule learner model which we denote as $\hat{\varphi}_\theta$. The learning objective of task L_i is defined as maximizing the score defined Eq.(4) for the samples in $\mathcal{D}_i$

$$\arg\max_{\theta} \sum_{\boldsymbol{x} \in \mathcal{D}_i} \text{score}(\boldsymbol{v}_{\boldsymbol{x}}, \alpha, \beta),$$
$$\text{where} \quad \alpha, \beta = \hat{\varphi}_\theta(\boldsymbol{v}_{\boldsymbol{x}} | \mathcal{X}_g, \mathcal{P}_g).$$

After training, we retrieve the discrete rules for later inference. This is done by taking the argmax of the attentions (α, β) over the selection dimensions. The resulting one-hot vectors then explicitly defines an entailment rule r. With a slight abuse of the notations, we denote this process as $r \leftarrow \varphi_\theta(r|\boldsymbol{x}, P_i, \mathcal{KG})$.

Query generation $\pi(r|\mathcal{R}_i)$: the agent builds the rule set $\mathcal{R}_i$ by collecting all the rules used in inferring the target samples in the warm-up set. With a limited budget, the query policy should prefer to pick rules that are more likely to improve the performance for evaluation. To keep the framework efficient, we resort to a simple approximation: we rank each of the unique rules with their frequencies in $\mathcal{R}_i$ and submit the top-B of them as queries. The assumption is that a frequently used rule is likely to apply to more data in the hidden set and can thus lead to better performance. As we will demonstrate it in the experiments, this intuition is indeed effective. In human experiments, the rules are translated into the natural language using a template-based method.

Synthetic oracle $O(r)$: due to the scale of the problem, full human evaluation is costly. We adopt the protocol in (Shen et al., 2019). On one hand, we implement a synthetic oracle that simulates the question asking process with the human. The oracle maintains a separate labeled set sampled from the target classes. During training, it evaluates the individual rules by applying them to the samples and computes the top-1 precision scores (i.e. P@1) with respect to the ground-truth labels. And the scores are returned to the agent as the oracle feedback. On the other hand, we verify the synthetic oracle in the offline setting: we collect a fixed set of rules generated by the model and evaluate them with human oracle. The results are then compared to those generated by the synthetic one for verification.

Labeling samples with rules $l(y|\boldsymbol{x}, \mathcal{R}_B, O)$: the rules are labeled with their precision scores which indicate the confidence of their outputs when they are applied to the samples. For sample labeling, we consider the rule as a noisy auto-labeling function, and the precision score as its random acceptance threshold. Formally, for each target sample $\boldsymbol{x} \in \mathcal{H}_i$, if $r(\boldsymbol{v}_{\boldsymbol{x}}) = 1$ for $r \leftarrow \varphi_\theta(r|\boldsymbol{x}, P_i, \mathcal{KG})$ and $r \in \mathcal{R}_B$, the label is accepted with the probability of $O(r)$. Recall that the entailment rules only recognize the positive samples, thus the estimated label $\hat{Y}_i$ contains only positive labels. To train the learner model, the agent augments $\hat{Y}_i$ with the same number of negative samples from the background set.

This process can potentially lead to (i) false negatives, where positive samples are not recognized because of the randomness, and (ii) false positives, where the rules over-generalize to other object classes. In the experiment, we will investigate the sample quality of the auto-labeling process and the impact of the false samples.

7. Experiments

In the experiments, we evaluate our method in a visual object classification scenario. For the background knowledge $\mathcal{KB}$, the agent is given a set of known object classes and common spatial and semantic relations. The agent is then tasked to

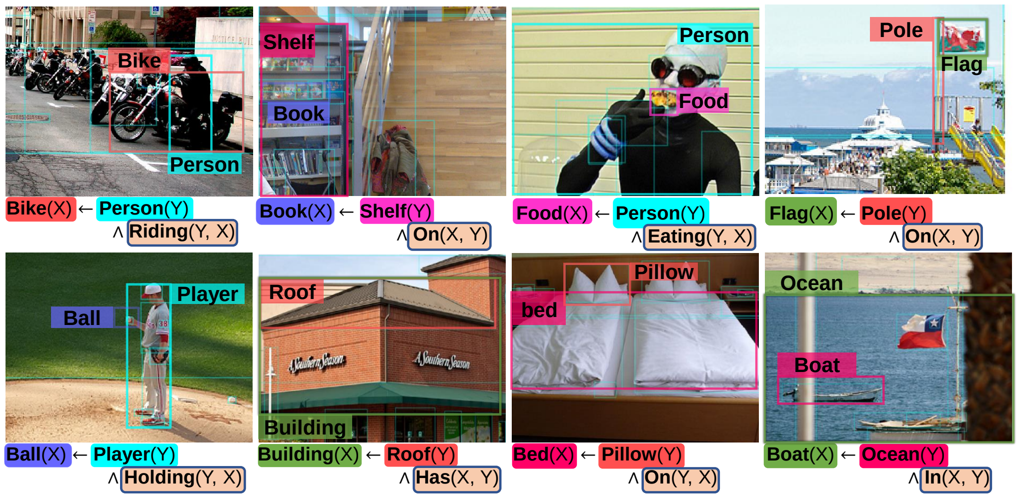
Bike
Person
Bike(X) ← Person(Y) ∧ Riding(Y, X)
Shelf
Book
Book(X) ← Shelf(Y) ∧ On(X, Y)
Person
Food
Food(X) ← Person(Y) ∧ Eating(Y, X)
Pole
Flag
Flag(X) ← Pole(Y) ∧ On(X, Y)
Player
Ball
Ball(X) ← Player(Y) ∧ Holding(Y, X)
Roof
Building
Building(X) ← Roof(Y) ∧ Has(X, Y)
Pillow
bed
Bed(X) ← Pillow(Y) ∧ On(Y, X)
Ocean
Boat
Boat(X) ← Ocean(Y) ∧ In(X, Y)

Figure 4: Showcases of using the learned rules to explain the target objects.

learn another set of new object classes where each comes with only a small set of labeled data.

7.1. Dataset

We conduct experiments on the Visual Genome dataset provided in (Krishna et al., 2016). The original dataset is highly noisy (Zellers et al., 2018). Here, we use its pre-processed version provided in the GQA dataset (Hudson & Manning, 2019). We further pre-process the relational data by filtering out the relation types that occur less than 1500 times. The remaining set consists of 31 relation categories. For each scene-graph, we filter out the isolated nodes that are not connected by those relations.

For the benchmarks, we rank all the object classes by their frequencies and keep the top 120 most frequent classes. We use the first 50 classes as background knowledge. All methods can access the entire labeled set of those classes for pre-training. Then all methods are evaluated on the rest of the 70 classes where only a small warm-up set is given for each class.

Throughout the training, the agent can access all the ground-truth relational data regardless of the class label of the object they connect to. In other words, the model can utilize the entire scene-graph information except for the node labels that belong to the 70 classes. Apart from the warm-up set, the model can access another set of labeled data with the same size as the warm-up set for validation. The rest of the labeled data of the target classes are split with a 90%/10% ratio into the hidden set and the testing set. Since the dataset only consists of positive samples, the hidden set is augmented with another set of objects randomly sampled from other target classes, such that the number of positive and negative samples are equal. All experiments are conducted on a machine with i7-8700K, 32G RAM and one GTX1080ti.

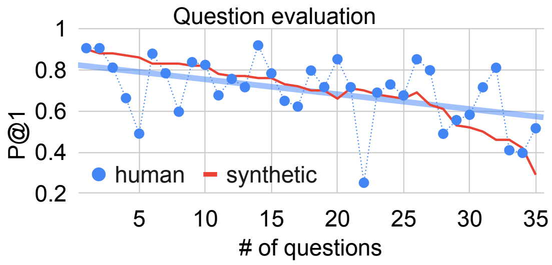

Figure 5: Human and synthetic oracle evaluation on the 35 FOL rules. Questions are sorted in descending order with respect to the oracle score.

7.2. Comparison of two types of supervision

Table 1: Average time taken in evaluating the individual samples, the commonsense questions and the amortized time of the questions.

Eval. type	Avg. time (s)	Std.(s)
Sample	3.7	1.8
Rule	8.3	3.1
Amortized	**0.03**	-

As one of the key motivations, we argue that evaluating the commonsense statements (i.e. FOL rules) is more efficient than sample-wise labeling since the human oracle does not need to go through the specific samples. To see this, we adopt a similar protocol as in (Shen et al., 2019). We quantify the cost of human supervision as the "wall clock time taken" for the oracle in labeling the samples or rules.

We collect 35 FOL rules learned from the warm-up set and translate them into natural language questions and put them into a survey form. We invite 25 graduate students to evaluate these statements. For each question, the participants choose from a 4-point scale that reflects how they think the rule is generally true or not (i.e. the precision score).

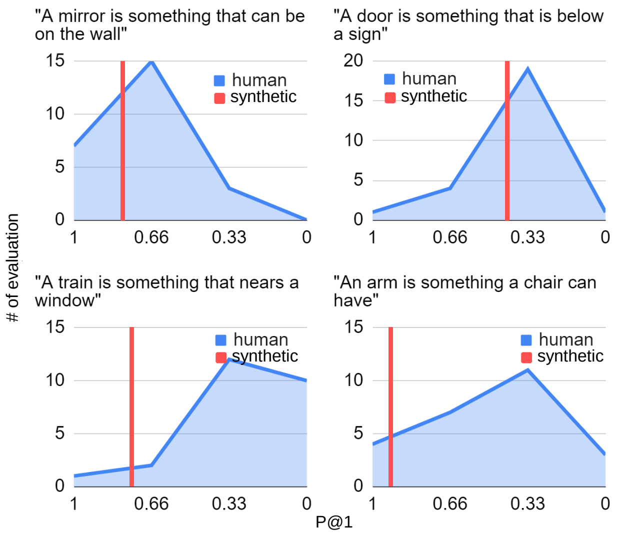

Figure 6: Human evaluation score distributions of 4 questions. The red bar corresponds the score of the synthetic oracle.

For sample-wise labeling, the participant is shown with 50 images randomly sampled from the target classes. Target objects are highlighted with the bounding boxes and the participant is asked to mark true or false on whether or not the image patch belongs to the target class.

We summarize the average labeling time of two types of supervisions in Table 1. Rule evaluation generally takes 8.3s which is 2x longer than the sample-wise labeling. In the experiment, this time is amortized as the rule applies to many samples. We can consider the "per-sample" cost by dividing the time with the total number of applicable samples. This leads to the 0.03s per-sample cost with rule evaluation as the supervision, which is 100x faster than the sample-wise counterpart.

7.3. Synthetic oracle vs. human

We summarize the synthetic oracle and human evaluation scores on the 35 rules in Figure 5. The 4-point scores in the human survey are converted to $\{1, 0.66, 0.33, 0\}$ of the P@1 scores and are averaged across participants for each question. Overall, the human evaluations tend to agree with that of the oracle: we fit the scores with a linear trend line (in light blue) and it is close to the score curve of the oracle. However, there are also several outliers where the two disagree significantly.

To see this, we illustrate the score distributions of specific questions in Figure 6. The oracle and the human agrees on the first two questions in the first row. However, for the questions related to `Train` and `Arm`, the oracle tends to over-estimate. We find many of them are due to the inherent bias presented in the dataset. For example, in real-life, a `Chair` does not usually have arms, but the `Arm` object is constantly labeled as part of the `Chair` object in the dataset, leading counter-intuitive statements. Here, we leave this problem for future investigation.

7.4. Visual Genome Benchmark

The goal of this benchmark is to evaluate the query-efficiency of our methods in learning new concepts that minimize the loss in Eq.(3).

Baselines: We consider 4 baselines from two domains: we compare 3 sample-wise active learning methods (i) **Random**, baseline method that submits random samples to the oracle as queries, (ii) **Entropy**, method that picks the sample with high entropy as the query and (iii) **LAL**, a meta active learning method that learns the sampling policy on-the-fly during the query process. We also compare against a meta-learning method (iv) **MAML** which learns a meta parameter initialization for the learner model that leads to the faster convergence in the future task.

Learner model: We apply all methods to two supervised learners: (i) **MLP** is a standard multi-layer fully connected classifier that takes the RCNN feature (provided in (Hudson & Manning, 2019)) of the object and outputs its class label. And (ii) **GCN** (Kipf & Welling, 2016) is the graph convolutional network that reads in the scene-graph and RCNN features as the node attribute and outputs the node embedding of the target object which is then fed into an MLP model to produce its class label. Our method uses the scene-graph on the active learning level. To make the comparison fair, we include this learner such that methods such as **MAML** and **LAL** can utilize and transfer the graph information as well. For **LAL**, the method builds on top of the internal features of a random forest learner model. To keep consistent with other methods, we save the samples that are selected during the meta-active learning phase and train the two learner models offline for evaluation.

Evaluation: For all experiments, we set the size of the warm-up set to 25 and vary the budget for each target class between $[1, 500]$. For each budget setting, We run all methods to learn the target 70 object classes. We compare the performance by showing the micro-averaged R@1 score of all classes, i.e. the top 1 recall of the learner model. Note that our method can learn rules at most the size of the warm-up set (i.e. one unique rule for each sample), thus for those budgets that exceed 25, we fill in queries from the **random** policy.

We summarize the results of **MLP** and **GCN** in Table 2 and Table 3. For **MLP** learner, our method outperforms the baselines significantly on small budgets. We find the improvement moving from budget 1 to 10 is small, and

Table 2: The micro-averaged R@1 scores of 5 methods on learning the 70 object classes with MLP learner and varied budget size.

Method w/ MLP	Budget					
	1	5	10	50	200	500
Random	0.47	0.47	0.5	0.52	0.57	0.6
Entropy	0.47	0.47	0.51	0.52	0.58	0.61
LAL	0.47	0.48	0.53	0.55	0.59	0.6
MAML	0.48	0.48	0.5	0.54	0.58	0.6
Ours	**0.56**	**0.57**	**0.58**	**0.6**	**0.62**	**0.63**

Table 3: The micro-averaged R@1 scores of 5 methods on learning the 70 object classes with GCN learner and varied budget size.

Method w/ GCN	Budget					
	1	5	10	50	200	500
Random	0.59	0.6	**0.62**	0.63	0.65	**0.67**
Entropy	0.59	0.59	**0.62**	0.64	0.65	**0.67**
LAL	0.58	0.58	0.61	0.64	**0.67**	**0.67**
MAML	0.59	0.61	0.61	0.63	0.64	0.65
Ours	**0.62**	**0.62**	**0.62**	**0.65**	0.65	0.65

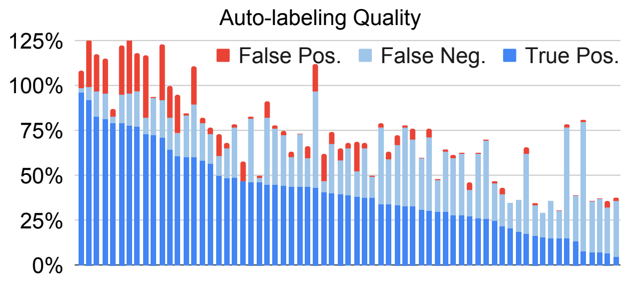

Figure 7: The ratio of true positive, false negative and false positive samples resulting from the auto-labeling for each target class. The ratio is calculated w.r.t the number of positive samples per class and can be greater than 100% if false positives are included.

this is due to that a large portion of the samples is usually covered by less than 3 rules. Additional rules typically cover rare patterns that only occur less than 50 times for each class. And by filling in random queries starting from budget 25, the performance improves at the rate similar to other baselines.

For **GCN** learner, the performance lead by our method is smaller, which is sensible because GCN is capable of generalizing over similar graph patterns. And since the relational data are all revealed in this benchmark, the model can generalize with significantly less node label data. On the other hand, we find some of the baselines outperform our method on a large budget, this is due the auto-labeling introduces the false positive samples and thus causes the performance to plateau.

Remarks: Comparing the number of the budget is technically unfair for the baselines because the rule carries more information than a single sample label. However, we note that the main motivation of this work is to propose an alternative supervision mechanism such that the human effort is reduced. In this way, as illustrated in section 7.2, being able to carry more information in a single query is by itself an advantage over the traditional paradigm.

7.5. Errors analysis on auto-labeling

As indicated in section 6, using the rules for auto-labeling can lead to both false negatives and false positives. We summarize these two types of errors when the agent runs with the budget of 5, and illustrate them for each class in Figure 7. Here, the ratio is measured with respect to the size of the positive sample per class and can be greater than 100% if many false positives are included. In general, we find the false positives counts for a small portion of the labeled data, which is also verified in the benchmark experiment.

In total, the rules recover 43.5% of the positive data. 23.4% of them are false negatives due to the random acceptance strategy, and the rest of the 23% does not apply to the learned rules. And with a comparable size of the negative sample pools, the agent only includes 6.7% of them as false positive samples.

8. Conclusion

One of the key characteristics of human learning is the ability to associate the new concept with the acquired ones and generalize over a few examples. In this work, we propose a novel active learning method to simulate this capability in the visual object classification scenario, i.e. the learning-by-asking-commonsense-question framework. Given a new task, the agent can generate commonsense questions by using the existing knowledge. In the experiments, we show that evaluating such questions leads to a significant reduction in human supervision.

References

Bachman, P., Sordoni, A., and Trischler, A. Learning algorithms for active learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 301–310. JMLR. org, 2017.

Chen, Z. and Liu, B. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 10(3):1–145, 2016.

Contardo, G., Denoyer, L., and Artières, T. A meta-learning approach to one-step active learning. *arXiv preprint arXiv:1706.08334*, 2017.

Das, R., Neelakantan, A., Belanger, D., and McCallum, A. Chains of reasoning over entities, relations, and text using recurrent neural networks. *arXiv preprint arXiv:1607.01426*, 2016.

Fang, M., Li, Y., and Cohn, T. Learning how to active learn: A deep reinforcement learning approach. *arXiv preprint arXiv:1708.02383*, 2017.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Gardner, M. and Mitchell, T. Efficient and expressive knowledge base completion using subgraph feature extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1488–1498, 2015.

Guu, K., Miller, J., and Liang, P. Traversing knowledge graphs in vector space. *arXiv preprint arXiv:1506.01094*, 2015.

Hudson, D. A. and Manning, C. D. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6700–6709, 2019.

Joshi, A. J., Porikli, F., and Papanikolopoulos, N. Multi-class active learning for image classification. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2372–2379. IEEE, 2009.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Konyushkova, K., Sznitman, R., and Fua, P. Learning active learning from data. In *Advances in Neural Information Processing Systems*, pp. 4225–4235, 2017.

Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.-J., Shamma, D. A., Bernstein, M., and Fei-Fei, L. Visual genome: Connecting language and vision using crowdsourced dense image annotations. 2016. URL `https://arxiv.org/abs/1602.07332`.

Lao, N. and Cohen, W. W. Relational retrieval using a combination of path-constrained random walks. *Machine learning*, 81(1):53–67, 2010.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., and Liu, S. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379*, 2015.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pp. 6467–6476, 2017.

Misra, I., Girshick, R., Fergus, R., Hebert, M., Gupta, A., and Van Der Maaten, L. Learning by asking questions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11–20, 2018.

Mitchell, T., Cohen, W., Hruschka, E., Talukdar, P., Yang, B., Betteridge, J., Carlson, A., Dalvi, B., Gardner, M., Kisiel, B., et al. Never-ending learning. *Communications of the ACM*, 61(5):103–115, 2018.

Munkhdalai, T. and Yu, H. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2554–2563. JMLR. org, 2017.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850, 2016.

Settles, B., Craven, M., and Ray, S. Multiple-instance active learning. In *Advances in neural information processing systems*, pp. 1289–1296, 2008.

Shen, T., Kar, A., and Fidler, S. Learning to caption images through a lifetime by asking questions. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 10393–10402, 2019.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.

Yang, F., Yang, Z., and Cohen, W. W. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems*, pp. 2319–2328, 2017.

Yang, J., Lu, J., Lee, S., Batra, D., and Parikh, D. Visual curiosity: Learning to ask questions to learn visual recognition. *arXiv preprint arXiv:1810.00912*, 2018.

Yang, Y. and Song, L. Learn to explain efficiently via neural logic inductive learning. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SJlh8CEYDB`.

Zellers, R., Yatskar, M., Thomson, S., and Choi, Y. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5831–5840, 2018.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3987–3995. JMLR. org, 2017.