frontiers in Robotics and AI

ORIGINAL RESEARCH
published: 18 November 2020
doi: 10.3389/frobt.2020.00122

Few-Shot Induction of Generalized Logical Concepts via Human Guidance

Mayukh Das[1]*, *Nandini Ramanan*[2], *Janardhan Rao Doppa*[3] and *Sriraam Natarajan*[2]

[1] Device Intelligence, Samsung R&D Institute India - Bangalore, Device Intelligence, Bangalore, India, [2] Erik Jonsson School of Engineering and Computer Science (ECS), The University of Texas at Dallas, ECS, Dallas, TX, United States, [3] School of Electrical Engineering & Computer Science (EECS), Washington State University, EECS, Pullman, WA, United States

We consider the problem of learning generalized first-order representations of concepts from a small number of examples. We augment an inductive logic programming learner with 2 novel contributions. First, we define a distance measure between candidate concept representations that improves the efficiency of search for target concept and generalization. Second, we leverage richer human inputs in the form of advice to improve the sample efficiency of learning. We prove that the proposed distance measure is semantically valid and use that to derive a PAC bound. Our experiments on diverse learning tasks demonstrate both the effectiveness and efficiency of our approach.

Keywords: cognitive systems, logics for knowledge representation, relational learning, knowledge representation and reasoning, human in the loop (HITL)

OPEN ACCESS

Edited by:
Amy Loutfi,
Örebro University, Sweden

Reviewed by:
Nicos Angelopoulos,
Cardiff University, United Kingdom
Leonardo Trujillo,
Instituto Tecnológico de Tijuana, Mexico

***Correspondence:**
Mayukh Das
mayukh.das@samsung.com

Specialty section:
This article was submitted to Computational Intelligence in Robotics, a section of the journal Frontiers in Robotics and AI

Received: *14 May 2020*
Accepted: *04 August 2020*
Published: *18 November 2020*

Citation:
Das M, Ramanan N, Doppa JR and Natarajan S (2020) Few-Shot Induction of Generalized Logical Concepts via Human Guidance. Front. Robot. AI 7:122. doi: 10.3389/frobt.2020.00122

1. INTRODUCTION

We study the case of learning from *few examples*, of which *one-shot* learning is a special case (Lake et al., 2015). We consider a challenging setting—that of learning explainable, decomposable, and generalizable (first-order) concepts from few examples. Plan induction becomes a special case where a generalizable plan is induced from a single (noise-free) demonstration. As an example, consider building a tower that requires learning *L-shapes* as a primitive. In our formulation, the goal is to learn a *L-shape* from a single demonstration. Subsequently, using this concept, the agent can learn to build a rectangular base (with 2 *L-shapes*) from another single demonstration and so on till the tower is fully built. Concept learning has been considered as problem solving by reflection (Stroulia and Goel, 1994), mechanical compositional concepts (Wilson and Latombe, 1994), learning probabilistic programs (Lake et al., 2015), etc. While successful, they are not considered in one-shot learning except with SVM (Tax, 2001), or with a neural network (Kozerawski and Turk, 2018).

Our work has two key differences. First, we aim to learn an "easily interpretable," "explainable," "decomposable," and "generalizable" concepts as first-order Horn clauses (Horn, 1951) (which are akin to If-Then rules). Second, and perhaps most important, we "do not assume the existence of a simulator (for plans) or employ a closed-world assumption" to generate negative examples. Inspired by Mitchell's (1997) observation of futility of bias-free learning, we employ domain expertise as inductive bias. The principle of structural risk minimization (Vapnik, 1999) shows how optimal generalization from extremely sparse observations is quite difficult. The problem is difficult in structured domains since most relations are false. Thus, few-shot induction of generalized logical concepts is challenging. We employ iterative revision of first-order horn clause theories using a novel scoring metric and guidance from a human. *We emulate a "student" who learns a generalized concept from an example provided by the "teacher," by both reflecting as well as asking relevant questions.*

We propose *Guided One-shot Concept Induction* (GOCI) for learning in relational domains[1]. GOCI builds upon an inductive logic program (ILP) learner (Muggleton, 1991) with two key extensions. First, a modified scoring function that explicitly computes distances between concept representations. We show the relation to *Normalized Compression Distance* (NCD) for plan induction settings. Consequently, we demonstrate that NCD is a valid distance metric. Second, we use domain knowledge from human expert as inductive bias. Unlike many advice taking systems that employ domain knowledge before training, GOCI identifies the relevant regions of the concept representation space and *actively solicits guidance* from the human expert to find the target concept in a sample-efficient manner. Overall, these two modifications allow for more effective and efficient learning using GOCI that we demonstrate both theoretically and empirically.

We make the following key contributions:

1. We derive a new distance-penalized scoring function that computes definitional distances between concepts, henceforth termed as "conceptual distance."
2. We treat the human advice as an inductive bias to accelerate learning. Our ILP learner actively solicits richer information from the human experts than mere labels.
3. Our theoretical analyses of GOCI prove that (a) our metric is indeed a valid distance, and (b) NCD between plans is a special case of our metric.
4. We show a PAC analysis of the learning algorithm based on Kolmogorov complexity.
5. We demonstrate the exponential gains in both sample efficiency and effectiveness of GOCI on diverse concept induction tasks with one or a few examples.

2. BACKGROUND AND RELATED WORK

Our approach to **Concept Learning** is closely related to Stroulia and Goel (1994)'s work, which learns logical problem-solving concepts by reflection. GOCI's scoring metric is more general and applicable to both concepts and plans and can be used for learning from a few examples. While we use discrete spatial structures as motivating examples, GOCI is not limited to discrete spaces, similar to Wilson and Latombe (1994)'s work. GOCI is also related in spirit to probabilistic (Bayesian) program induction for learning decomposable visual concepts (Lake et al., 2015), which illustrates how exploiting decomposability is more effective. Our approach leverages not only decomposability but also implicit relational structure.

2.1. One/Few-Shot Learning and Theory Induction

Our problem setting differs from the above in that it requires learning from sparse examples (possibly one). Lake et al. (2015) propose a one-shot version of Bayesian program induction of visual concepts. There is also substantial work on one/few-shot learning (both deep and shallow) in a traditional classification setting (Bart and Ullman, 2005; Vinyals et al., 2016; Wang et al., 2018), most of which either pre-train with gold-standard support example set or sample synthetic observations. We make no such assumptions about synthetic examples. ILP (Muggleton, 1991) inductively learns a logical program (first-order theory) that covers most of the positive examples and few of the negative examples by effectively employing background knowledge as search bias. In concept learning, generalization is typically performed as a search through space of candidate inductive hypotheses by (1) structuring, (2) searching, and (3) constraining the space of theories. FOIL (Quinlan, 1990) is an early noninteractive learner with the disadvantage that it occasionally prunes some uncovered hypotheses. This is alleviated in systems like FOCL by introducing language bias in the form of user-defined constraints (Pazzani, 1992). With *Interactive ILP*, learner could pose questions and elicit expert advice that allows pruning large parts of search space (Sammut and Banerji, 1986; Rouveirol, 1992). To incorporate new incoming information, ILP systems with *theory revision* incrementally refine and correct the induced theory (Sammut and Banerji, 1986; Muggleton, 1988). While GOCI is conceptually similar to ALEPH (Srinivasan, 2007), it learns from a few examples and actively acquires domain knowledge by interacting with human expert incrementally.

2.2. Knowledge-Guided Learning

Background knowledge in ILP is primarily used as search bias. Although the earliest form of knowledge injection can be found in explanation-based approaches (Shavlik and Towell, 1989), our work relates to preference-elicitation framework (Braziunas and Boutilier, 2006), which guides learning via human preferences as an inductive bias. Augmented learning with domain knowledge as an inductive bias has long been explored across various modeling formalisms, including traditional machine learning (Fung et al., 2003), probabilistic logic (Odom et al., 2015), and planning (Das et al., 2018). Our human-guided GOCI learner aims to extend these directions in the context of learning generalizable complex concepts from a few examples(including plans). Similar problem setting of concept learning from incomplete/sparse observations has also been explored in the cognitive science paradigm via explanation-based inductive program synthesis (Flener, 1997; Kitzelmann and Schmid, 2006).

The idea of augmented learning with human guidance/knowledge has also been extensively studied in the context of evolutionary computation. Interactive evolutionary systems (Eiben and Smith, 2015) use expert guidance to emulate a holistic fitness function that would otherwise depend on a very restricted pre-defined fitness model. The potential richness of such knowledge can be leveraged in not just evolutionary parent selection but can also optimize other parameters that leads to faster convergence, especially in mutations (Wendt et al., 2010). ILP has been shown to be conceptually similar to mutative EA in the context of program induction (Wong and Leung, 1997) and hence knowledge-guided mutations are related to knowledge augmented search in ILP. Thus, in our problem setting, the interaction module that seeks human guidance to select the most useful constraints (detailed in section 3.2.3) is similar in spirit to interactive (knowledge guided) evolutionary mutation process.

[1]Our algorithm can learn from one (few) example(s). We specify the number of examples in our evaluations.

However, our underlying search strategy and optimization is based on ILP.

3. GUIDED ONE-SHOT CONCEPT INDUCTION

We are inspired by a teacher (human) and student (machine) setting in which a small number of demonstrations are used to learn generalized concepts (Chick, 2007). Intuitively, the description provided by a human teacher tends to be modular (can have distinct logical partitions), structured (entities and relations between them), and in terms of known concepts. Hence, a vectorized representation of examples is insufficient. We choose a logical representation, specifically a "function-free restricted form of first-order logic (FOL)" that models structured spaces faithfully.

Given: A set of "facts" or assertions, that is, a set of ground literals (or trajectories) describing 1 (or few) instance(s) of an unknown concept, availability of an expert to provide guidance and a knowledge-base of known concepts.
To Do: Learning a representation, by inducing a first-order logic program, of the given concept that optimally generalizes the given instance(s) effectively.

The input to GOCI is the description of the instances(s) of a concept that the human teacher provides. An example is, thus, conjunction of a set of ground literals (assertions). The output of GOCI is a least general generalization (LGG) horn clause from the *input* example(s).

3.1. Concept Representation

Consider the following example input to the GOCI framework. The input here is an instance of the structural concept $\mathbb{L}$ (illustrated in **Figure 1**).

Example 1. *An instance in a minecraft domain can be a* $\mathbb{L}$ *with dimensions height* $= 5, base = 4$ (***Figure 1***). $\mathbb{L}(S), Height(S, 5), Base(S, 4)$, *s is the concept identifier and may be described as conjunction of ground literals,*

$\texttt{Row(A)} \wedge \texttt{Tower(B)} \wedge \texttt{Width(A,4)} \wedge \texttt{Height(S,5)} \wedge \texttt{Base(S,4)} \wedge \texttt{Contains(S,A)} \wedge \texttt{Contains(S,B)} \wedge \texttt{Height(B,4)} \wedge \texttt{SpRel(B,A,'NWTop')}$,

which denotes $\mathbb{L}$ *as composition of a "Row" of* $w = 4$ *and a "Tower" of* $h = 4$ *with appropriate literals describing the scenario (**Figure 1**, left). As a special case, under partial or total ordering assumptions among the ground literals, an input instance can represent a plan demonstration.*

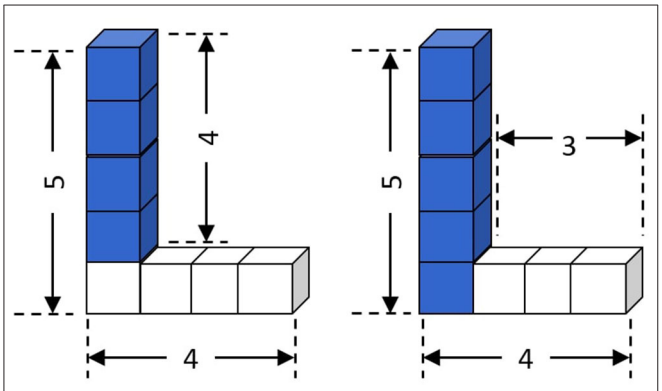

FIGURE 1 | Concept $\mathbb{L}$ ($base = 4, height = 5$), described as composition of a **Tower** and a **Row**.

We aim to learn the optimally generalized (decomposable) representation of the concept ($\mathbb{L}$ in the context of the aforementioned example) referred by the one/few instances that were passed to GOCI as input. Before further discussion on the learning such a generalized (decomposable) representation let us first define formally what a concept representation signifies in our setting.

Definition 1. *Concepts in* GOCI *are represented as horn clause theories. A theory T is defined as,* $T = \mathcal{C}(s^k \ldots) :- \bigvee \left[\wedge_{i=1}^{N} f_i(t_1, \ldots, t_j)\right]$, *where the body* $\wedge_{i=1}^{N} f_i(t_1, \ldots, t_j)$ *is a conjunction of literals indicating known concepts or relationships among them, such that any* $t_j \in V \cup \{s^k\} \cup C$ *where V is the set of all logical variables in the clause, C is the set of constants in the domain of any logical variable. The head* $\mathcal{C}(s^k \ldots)$ *identifies a target concept, and the terms* $\{s^k\}$ *are logical variables that denote the parameters of the concept assuming there are* $k = \{1, \ldots, K\}$ *parameters including the identifier to the given instance of the concept. Since a concept can be described in multiple ways (**Figure 1**), the final theory will be a disjunction over clause bodies with the same head. A (partial) instantiation of a theory T is denoted as* T/θ.

Note that these definitions allow for the reuse of concepts, potentially in a hierarchical fashion. We believe that this is *crucial in achieving human-agent collaboration.*

Example 2. ***Figure 1*** *illustrates an instance of the concept* $\mathbb{L}$ *that can be described in multiple ways. A possible one is,*

$$\begin{aligned}\mathbb{L}(s) :- [&\texttt{Height(s,h}_\texttt{s}\texttt{)}, \texttt{Base(s,w}_\texttt{s}\texttt{)}, \texttt{Contains(s,a)},\\ &\texttt{Contains(s,b)}, \texttt{Row(a)}, \texttt{Tower(b)},\\ &\texttt{Width(a,w}_\texttt{a}\texttt{)}, \texttt{Height(b,h}_\texttt{b}\texttt{)}, \texttt{Equal(w}_\texttt{s}\texttt{,w}_\texttt{a}\texttt{)},\\ &\texttt{Sub(h}_\texttt{b}\texttt{,h}_\texttt{s}\texttt{,1)}, \texttt{SpRel(b,a,''NWTop'')}]\\ \bigvee\ [&\texttt{Height(s,h}_\texttt{s}\texttt{)}, \texttt{Base(s,w}_\texttt{s}\texttt{)}, \texttt{Contains(s,a)},\\ &\texttt{Contains(s,b)}, \texttt{Row(a)}, \texttt{Tower(b)},\\ &\texttt{Width(a,w}_\texttt{a}\texttt{)}, \texttt{Height(b,h}_\texttt{b}\texttt{)}, \texttt{Equal(h}_\texttt{s}\texttt{,h}_\texttt{b}\texttt{)},\\ &\texttt{Sub(w}_\texttt{a}\texttt{,w}_\texttt{s}\texttt{,1)}, \texttt{SpRel(b,s,''W'')}]\end{aligned}$$

The generalization must be noted. The last argument of the `SpRel()` is a constant, as only this particular spatial alignment is appropriate for the concept of the $\mathbb{L}$ structure. Although the input is a single instance (Example 1), GOCI should learn a generalized representation such as Example 2. Another

interesting aspect is the additional constraints: `Equal(X,Y)` and `Sub(X,Y,N)`. While such predicates are a part of the language, they are not typically described directly in the input examples. However, they are key to generalization, since they express complex interactions between numerical (or non-numerical) parameters. Also note that the head predicate of the clause could have been designed differently as per Definition 1. For instance, in case of Example 2, the head predicate could have folded in the dimensional parameters—$\mathbb{L}(s, h_s, w_s)$. However, the number of such dimensional parameters can vary across different concepts. Hence to maintain generality of representation format during implementation, we push the dimensional parameters of the learnable concept into the body of the clause.

A specific case of our concept learning (horn clause induction) framework could be plan induction from sparse demonstrations. This can be achieved by specifying time as the last argument of both the state and action predicates. Following this definition, we can allow for plan induction as shown in our experiments. Our novel conceptual distance is clearer and more intuitive in the case of plans as can be seen later.

Definition 2 (Decomposable:). *A concept $\mathcal{C}$ is decomposable if it is expressed as a conjunction of other concepts, and one or more additional literals to model the interactions. $\mathcal{C} \Leftarrow (\bigwedge_i \mathcal{C}'_i) \wedge (\bigwedge_j B_j)$. Here $\mathcal{C}'_i$ are literals that represent other concepts that are already present in the knowledge base of the learner and B_j are literals that either describe the attributes of $\mathcal{C}'_i$ or interactions between them.*

Decomposable allows for an unknown concept to be constructed as a composition of other known concepts. GOCI learns the class of decomposable concepts since it is intuitive for the "human teacher" to describe. Decomposable concepts faithfully capture the *modular* and *structured* aspect of how humans would understand and describe instances. It also allows for a hierarchical construction of plans.

Example 3. *Following the Minecraft structure described in 2, note how $\mathbb{L}$ is described with already known concepts $\mathcal{C}'_1 = Row()$ and $\mathcal{C}'_2 = Tower()$ and the other literals such as $Height(b, h_b), SpRel(b, a, \text{"NWTop"}), \ldots \in \{B_j\}$, that is, they describe the parameters of the known concepts or interactions between them. Note that known concepts in the knowledge base could have been manually coded in by experts or learned previously and are essentially represented in the same way. For instance, Row() can be encoded as recursive the clause program representing a composition of one block and one unit shorter row,*

$$Row(r) : -[Width(r, w_r), Block(a), Row(b), Width(b, w_b), SpRel(a, b, \text{"East"}), Sub(w_b, w_r, 1)] \bigvee [Width(r, w_r), Equal(w_r, 1), Block(a)]$$

Tower() could also be defined in the knowledge base in the same way. When the optimally general representation of the concept $\mathbb{L}$ is learned that is persisted in the knowledge base as well, such that more complex concepts can be represented by decomposing into $\mathbb{L}$ and other known concepts.

An obvious question that arises here is why $\{B_j\} \not\subseteq \{\mathcal{C}'\}$? that is, why can the other literals not be treated similarly as a part of the knowledge base of known concepts? Ideally, that would be correct. However, that will also cause infinite levels of concept definitions, which cannot be implemented in practice. Additionally, following the paradigm of a student–teacher scenario, it can always be assumed that the student has prior understanding of many concepts from outside the current system. Thus, we can safely assume, without loss of generality, that set of literals $\{B_j\}$ are implicitly understood and defined as a part of the framework itself. This argument applies to the semantics of the "constraint predicates" (described later) as well.

Finally, before we discuss the details of the learning methodology, let us briefly look into a motivating, and presently relevant, real-world scenario that represents our problem setting.

Example 4. *Consider a decision support AI system for resource planning and management in hospitals as illustrated in **Figure 2**. The AI agent forecasts the need for increased resources in the*

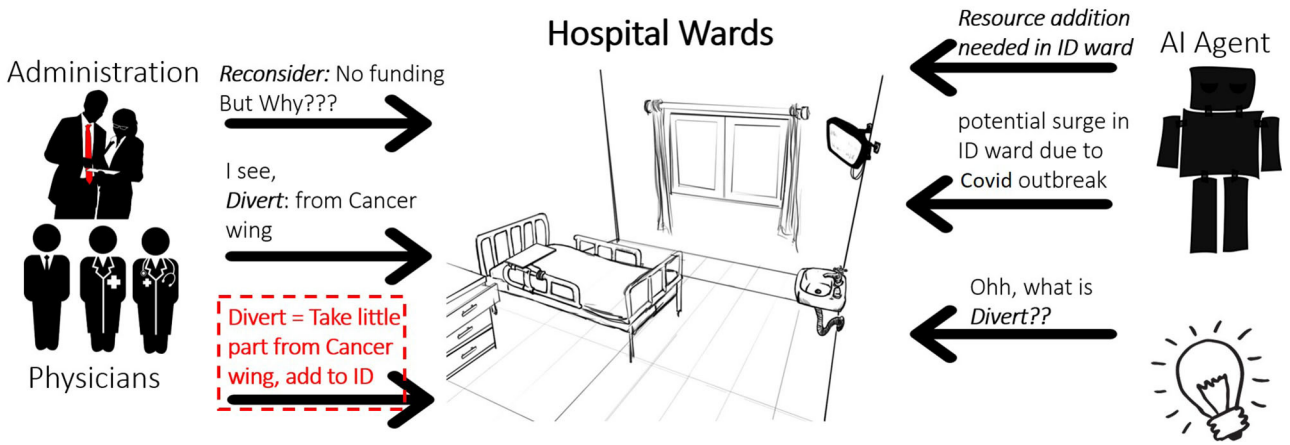

FIGURE 2 | A motivating real-world scenario for concept induction. The concept learnt by the AI agent is "Divert()" .

infectious diseases (ID) ward, given the early signs of an outbreak of some disease such as Covid-19 or Ebola, and a potential spike in ID ward admissions. However, as noted by the administrators and/or physicians there is not enough budget to procure additional resources for ID ward. But the problem is quite critical and needs to be solved. So the human teacher (administrators in this case) teaches the AI agent the concept of "Divert" -ing resources from Cancer ward since cancer ward admissions are usually stable and does not have spikes. The AI agent is hence expected to learn a generalized representation of the concept of divert such that it may be applied later for other wards or for other tasks and furthermore in a "decomposable" fashion. For instance, "Divert" may be learned as a clause program such as,

$$\begin{aligned}\texttt{Divert}(\texttt{R}, \texttt{qty}_{\texttt{R}}) :- &\texttt{To}(\texttt{R}, \texttt{Loc}_{\texttt{dest}}), \texttt{AcquireFrom}\\ &(\texttt{R}, \texttt{Loc}^1_{\texttt{source}}, \texttt{qty}_1,),\\ &\texttt{AcquireFrom}(\texttt{R}, \texttt{Loc}^2_{\texttt{source}}, \texttt{qty}_2),\\ &\texttt{AssignTo}(\texttt{R}, \texttt{Loc}_{\texttt{dest}}, \texttt{qty}_{\texttt{dest}}),\\ &\texttt{sum}(\texttt{qty}_1, \texttt{qty}_2, \texttt{qty}_{\texttt{dest}})\end{aligned}$$

Obviously, the above representation assumes that concepts such as "`AcquireFrom()`" are known concepts, either implicitly defined inside the learning framework or its explicit representation has been learned and persisted inside the knowledge base in the past.

The above example is solely to motivate the potential impact of our problem setting and the proposed solution. For an explanation of different components and aspects of GOCI, we refer to the much simpler and unambiguous structural example outlines earlier ($\mathbb{L}$).

3.2. Methodology

3.2.1. Search

ILP systems perform a greedy search through the space of possible theories. Space is typically defined, declaratively, by a set of mode definitions (Muggleton, 1995) that guide the search. We start with the most specific clause (known as a bottom clause) (Srinivasan, 2007) from the ground assertions and successively add/modify literals that might improve a rule that best explains the domain. Typically, the best theory is the one that covers the most positive and least negative examples. Thus, it optimizes the likelihood of a theory T based on the data. We start with a bottom clause and variablize the statements via anti-substitution. Variabilization of T is denoted by $\theta^{-1} = \{a/x\}$, where $a \in consts(T)$, $x \notin vars(T)$. That is, anti-substitution θ^{-1} is a mapping from occurrences of ground terms in T to new or existing logical variables.

Evaluation Score: We redesign the ILP scoring (e.g., ALEPH's compression heuristics) as:

- The user-provided advice forces the learner to learn longer theory, hence the *search space can be exponentially large*. Thus, modes alone are not sufficient as the search bias.
- There is only one (a few) positive training example(s) to learn from and *many possible rules can accurately match the training example*. Coverage-based scores fail.

Most learners optimize some form of likelihood. For a candidate theory T, likelihood given data $\mathbb{D}$ is $LL(T) = \log P(\mathbb{D}|T)$ (i.e., coverage). To elaborate further, in most classification tasks in discrete domains (with categorical/ordinal feature and target variables), goodness of fit of candidate models is achieved via the measure of how well the candidate models explain (or cover) the given data, that is, a good model is the one that will predict positive class for maximum possible positive examples and for minimum possible negative examples. This measure is expressed as likelihood of the data given a candidate model. In GOCI, we have one (at most few) positive example(s). Coverage will not suffice. Hence, we define a modified objective as follows.

$$T^* = \underset{T \in \tau}{arg\,min} \left(-LL(T) + D(T/\theta_X, X)\right) \tag{1}$$

where T^* is the optimal theory, τ is the set of all candidate theories, and D is the conceptual distance between the instantiated candidate theory T/θ_X and the original example X. Recall that a theory $\mathbb{T}$ is a disjunction of horn clause bodies (or conjunction of clauses).

3.2.2. Distance Metric

Conceptual distance, $D(T/\theta_X, X)$, is a penalty in our objective. The key idea is that any learned first-order horn clause theory must recover the given instance by equivalent substitution. However, syntactic measures, such as edit distance, are not sufficient since changing even a *single literal*, especially, literals that indicate interconcept relations, could potentially result in a completely different concept. For instance, in blocks-world, the difference between a block being in the middle of a row and one at the end of the row can be encoded by changing one literal. Hence, a more sophisticated semantic distance such as conceptual distance is necessary (Friend et al., 2018). However, such distances require deeper understanding of the domain and its structure.

Our solution is to employ *interplan distances*. Recall that the concepts GOCI can induce are decomposable and, hence, are equivalent to parameterized planning tasks. One of our key contributions is to exploit this equivalence by using a domain-independent planner to find grounded *plans* for both the theory learned at a particular iteration i, T_i and the instance given as input, X. We then compute the normalized compression distance (NCD) between the plans.

NCD: Goldman and Kuter (2015) proved that NCD is arguably the most robust interplan distance metric. NCD is a reasonable approximation of *Normalized Information Distance*, which is not computable (Vitányi et al., 2009). Let the plans for T_i/θ_X and X be π_T and π_X. To obtain NCD, we execute string compression (lossy or lossless) on each of the plans as well as the concatenation of the two plans to recover the compressed strings C_T, C_X, and $C_{T,X}$, respectively. NCD between the plans can be computed as,

$$NCD(\pi_T, \pi_X) = \frac{C_{T,X} - \min(C_T, C_X)}{\max(C_T, C_X)} \tag{2}$$

The conceptual distance between a theory T and X is the NCD between the respective plans, $D(T/\theta_X, X) = NCD(\pi_T, \pi_X)$.

This entire computation is performed by the *conceptual distance calculator* as shown in **Figure 3**.

Observations: **(1)** Conceptual distance as a penalty term in the *LL* score ensures that the learned theory will correctly recover the given example/demonstration. **(2)** $D(T/\theta_X, X)$ generalizes to the *Kolmogorov–Smirnov* statistic between two target distributions if we induce probabilistic logic theories. We prove these insights theoretically.

3.2.3. Human Guidance

The search space in ILP is provably infinite. Typically, language bias (modes) and model assumptions (closed world) are used to prune the search space. However, it is still intractable with one (or few) examples. So, we employ human expert guidance as constraints that can directly refine an induced theory, acting as a strong inductive bias. Also, we are learning decomposable concepts (see Definition 2). This allows us to exploit another interesting property. Constraints can now be applied over the attributes of the known concepts that compose the target concept, or over the relations between them. Thus, GOCI directly includes such constraints in the clauses as literals (see Example 2). Though such constraint literals come from the pre-declared language, they are not directly observed in the input example(s). So an ILP learner will fail to include such literals.

If the human inputs (constraints) are provided upfront before learning, it can be wasteful/irrelevant. More importantly, it places an additional burden on the human. To alleviate this, *our framework explicitly queries for human advice on the relevant constraint literals, which are most useful.* Let $\mathbb{U}$ be a predefined library of constraint predicates in the language, and let $\mathcal{U}() \in \mathbb{U}$ be a relevant constraint literal. Human advice $\mathcal{A}$ can be viewed as a preference over the set of relevant constraints $\{\mathcal{U}()\}$. If $U_{\mathcal{A}}$ denotes the preferred set of constraints, then we denote the theory having a preferred constraint literal in the body of a clause as $\tau_{\mathcal{A}}$. *(For instance, as per Example 2* GOCI *queries "which of the two sampled constraints* $\texttt{Sub}(\texttt{h}_\texttt{b}, \texttt{h}_\texttt{s}, \texttt{1})$ *&* $\texttt{Greater}(\texttt{h}_\texttt{b}, \texttt{h}_\texttt{s})$ *is more useful." Human could prefer* $\texttt{Sub}(\texttt{h}_\texttt{b}, \texttt{h}_\texttt{s}, \texttt{1})$*, since it subsumes the other.)* The scoring function now becomes:

$$T^* = \underset{T \in \tau}{arg\,min} \left(-LL(T) + D(T/\theta_X, X)\right) \; : \tau \subseteq \{\tau_{\mathcal{A}}\} \qquad (3)$$

Thus, we are optimizing the constrained form of the same objective as Equation (1), which aims to prune the search space. This is inspired by advice elicitation approaches (Odom et al., 2015). While our framework can incorporate different forms of advice, we focus on preference over constraints on the logical variables. The formal algorithm, described next, illustrates how we achieve this via an iterative greedy refinement (**Figure 3**, query-advice loop shown in left).

3.3. The GOCI Algorithm

Algorithm 1 outlines the GOCI framework. It initializes a theory T_0 by variablizing the *"bottom clause"* obtained from X and background knowledge [`lines 3 and 5`]. Then it performs a standard ILP search (described earlier) to propose a candidate theory [`line 6`]. This is followed by the guided refinement steps, where constraint literals are sampled (parameter tying guides the sampling) and the *human teacher* is queried for preference over them, such that the candidate theory can be modified using preferred constraints [`lines 7-9`]. The function NCD() performs the computation of the conceptual distance by first grounding the current modified candidate theory T' with the same parameter values as the input example X, then generating grounded plans and finally calculating the normalized compression distance between the plan strings (as shown in **Figure 3** and Equation 2) [`line 10`]. The distance-penalized negative log-likelihood score is estimated and minimized to find the best theory at the current iteration [`lines 11-14`], which is then used as the initial model in the next iteration. This process is repeated either until convergence (no change in induced theory) or maximum iteration bound (L).

Connection to plan induction: Evaluation, both in traditional ML and ILP, generally predicts the value of $\hat{y}_X$ for a test instance X represented as a fixed (ML) or arbitrary (ILP) length feature vector. In GOCI, however, the notion of evaluation of an instance X depends on successful construction of a valid/correct

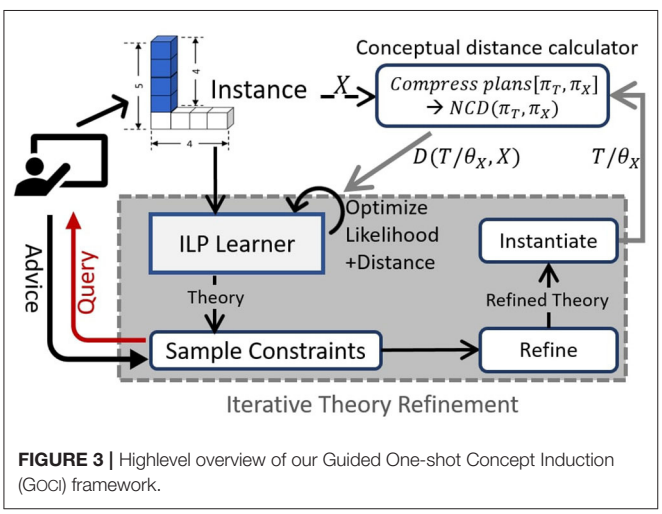

FIGURE 3 | Highlevel overview of our Guided One-shot Concept Induction (GOCI) framework.

Algorithm 1: Guided one-shot concept induction.

1: **procedure** GOCI(Instance X)
2: **Initialize:** Set Iteration $\ell \leftarrow 1$
3: **Initialize:** Bootstrap theory $T_0 \leftarrow X/\theta^{-1}$
4: **repeat**
5: Use $T_{\ell-1}$ as initial model
6: Candidate theory $T_\ell \leftarrow$ SEARCH($T \in \tau | T_{\ell-1}$)
7: **Sample** applicable constraints $\mathcal{U} \in \mathbb{U}$
8: $\mathcal{U}_{\mathcal{A}} \leftarrow$ QUERY($human, \mathcal{U}$)
9: $T' \leftarrow T_\ell \oplus \mathcal{U}_{\mathcal{A}}$ ▷ $\forall\, \mathcal{U}_{\mathcal{A}} \in \mathcal{A}$
10: $D_\ell(T'/\theta_X, X) \leftarrow$ NCD($\pi_{T'/\theta_X}, \pi_X$)
11: Score $\mathcal{S}_\ell \leftarrow \left(-LL(T') + D(T'/\theta_X, X)\right)$
12: **if** $S_\ell < S_{\ell-1}$ **then** ▷ minimize
13: Retain T': Update $T_\ell = T'$
14: **end if**
15: **until** $\ell \leq L$ OR $T_\ell = T_{\ell-1}$
16: **end procedure**

plan π_X (**Figure 4**). Thus, while learning, most research aim to maximize coverage of positive instances E^+ ($\max P(\hat{y}_x = true|y_x \in E^+)$) and minimize coverage of negatives E^-, [i.e., $\min P(\hat{y}_x = true|y_x \in E^-)$]. GOCI evaluates a candidate concept representation by allowing the agent to realize that concept—by computing a valid plan for the goal/task implied by the instance x. This is akin to plan induction, since we are learning parameterized plan for realizing the concept as a surrogate for the concept itself. Additionally, planning has long been shown to be conceptually same as logic programming (Preiss and Shai, 1989) and hence induction of logic programs (theories) is the same as plan induction where the examples are trajectories (plan traces) in this case.

3.4. Theoretical Analysis

3.4.1. Validity of Distance Metric

NCD $\delta(x, y)$ between two strings x and y is provably a valid distance metric (Vitányi et al., 2009): $\delta(x, y) = \frac{\max K(x|y), K(y|x)}{\max K(x), K(y)}$, where $K(x)$ is the Kolmogorov complexity of a string x and $K(x|y)$ is the conditional Kolmogorov complexity of x given another string y. NCD is a computable approximation of the same $[D(x, y) \approx \delta(x, y)]$. Thus, we just verify if δ is a correct conceptual distance measure. Let T_Y and T_Z be two theories, with same parameterizations (i.e., same heads). Let T_Y/θ and T_Z/θ be their groundings with identical parameter values θ. Our learned theories are equivalent to planning tasks. Assuming access to a planner $\Pi()$ which returns $Y = \Pi(T_Y/\theta)$ and $Z = \Pi(T_Z/\theta)$, the two plan strings with respect to the instantiations of concepts are represented by T_Y and T_Z, respectively.

Proposition 1 (Valid Conceptual Distance)**.** *Normalized information distance $\delta(Y, Z)$ is a valid and sound conceptual distance measure between T_Y and T_Z, that is, $\delta(Y, Z) = 0$ iff the concepts represented by T_Y and T_Z are equivalent.*

Proof Sketch for Proposition 1: Let T_Y and T_Z be 2 induced consistent first-order Horn clause theories, which may or may not represent the same concept. Let θ be some substitution. Now let T_Y/θ and T_Z/θ be the grounded theories under the same substitution. This is valid since we are learning horn clause theories with the same head, which indicates the target concept being learned. As explained in the paper, a theory is equivalent to a planning task. We assume access to a planner $\Pi()$, and we get plan strings $Y = \Pi(T_Y/\theta)$ and $Z = \Pi(T_Z/\theta)$ with respect to the planing tasks T_Y/θ and T_Z/θ.

Friend et al. (2018) proved that *Conceptual Distance* is the step distance between two consistent theories in a cluster network $(\mathbb{T}, \rightleftarrows, \sim)$, where $\mathbb{T}$ is the class of consistent theories, $\rightleftarrows$ is the definitional equivalence relation (equivalence over bidirectional concept extensions) and $\sim$ implies symmetry relation. We have shown in the paper that, given the class of concepts we focus on, *a concept is a planning task*.

Let there be a theory T^*, which represents the optimal generalization of a concept C. If step distance $\langle T_Y, T^* \rangle = 0$ in a cluster network and $\langle T_Z, T^* \rangle = 0$, then $\langle T_Y, T_Z \rangle = 0$, that is, they represent the same concept C and they are definitionally equivalent $T_Y \rightleftarrows T_Z$. Thus, both T_Y/θ and T_Z/θ will generate the same set of plans as T^*, since they will denote the same planning tasks (by structural induction). Thus,

$$T_Y \rightleftarrows T_Z \iff [\Pi(Y) \cap \Pi(Z) = \Pi(Y) = \Pi(Z)] \quad (4)$$

up to equivalence of partial ordering in planning. Let $\pi^*()$ be a minimum length plan in a set of plans $\Pi()$. Let y and z be strings indicating plans $\pi^*(Y)$ and $\pi^*(z)$ ignoring partial order. If $\Pi(Y) = \Pi(Z)$, then $\pi^*(Y) = \pi^*(z)$. Hence, the conditional Kolmogorov complexities $K(y|z)$ and $K(z|y)$ will both be set to 0, if the strings x and y are equivalent (ignoring partial ordering). This is based on the principle that if they are equivalent, then a Universal prefix-Turing machine will recover one string given the other in 0 steps.

$$\therefore \frac{max\left(K(y|z), K(z|y)\right)}{max\left(K(y), K(z)\right)} = 0 = \delta(Y, Z)$$

Proposition 2 (Generalization to Kolmogorov–Smirnov)**.** *In generalized probabilistic logic, following Vitányi (2013), $\delta(Y, Z)$ corresponds to 2-sample **Kolmogorov–Smirnov statistic** between two random variables T_Y/θ and T_Z/θ with distributions P_{T_Y} and P_{T_Z}, respectively, $[v(T_Y, T_Z) = \sup_{\theta \in \mathcal{F}} |F_{T_Y}(\theta) - F_{T_Z}(\theta)|]$, where $F_T()$ is the cumulative distribution function for P_T and $\sup_{\theta \in \mathcal{F}}$ is the supremum operator. In a deterministic setting, δ is a special case of the Kolmogorov–Smirnov statistic v, $\delta(Y, Z) \preceq v(F_{T_Y}, F_{T_Z})$.*

Proof Sketch for Proposition 2: This can be proved by considering the connection between NID and the distributions

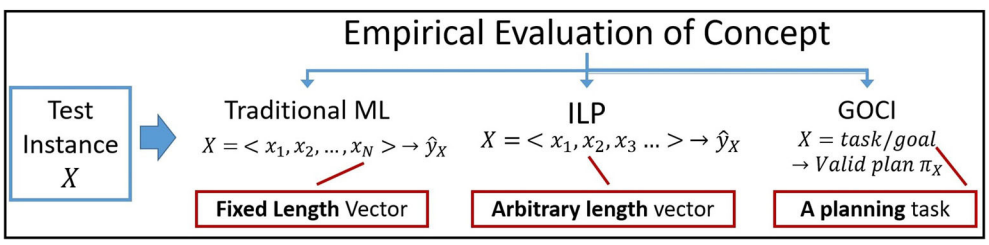

FIGURE 4 | Difference in evaluation of a concept instance across different learning paradigms.

induced by the concept classes we are learning. NID is defined as $\delta(x,y) = \frac{\max K(x|y),K(y|x)}{\max K(x),K(y)}$, where, $K(a|b)$ is the conditional Kolmogorov complexity of a string a, given b. There is no provable equivalence between Kolmogorov complexity and traditional notions of probability distributions.

However, if we consider a *reference* universal semi-computable semi-probability mass function $\mathbf{m}(x)$, then there is a provable equivalence $-\log \mathbf{m}(x) = K(x) \pm O(1)$. Similarly for conditional Kolmogorov complexity, by Conditional Coding Theorem, $-\log \mathbf{m}(y|x) = K(y|x) \pm O(1)$ (Vitányi, 2013). By definition,

$$m(y|x) = \sum_{j \geq 1} 2^{-K(j)-c_j} P_j(y|x)$$

where $c_j > 0$ are constants and $P_j(y|x)$ is the lower semi-computable conditional. A lower semi-computable semi-probability conditional mass function is based on the string generating complexity of a Universal prefix-Turing machine. Thus, $m(y|x)$ is greater than all the lower semi-computable. Note that our compressed plans are equivalent to a string generated by Universal prefix-Turing machines. The conditional case implies, if a compressed plan string x is given as an auxiliary prefix tape, how complex it is to generate compressed string $y = \theta$.

Given two grounded theories T_Y/θ and T_Z/θ, let $P_{T_Y/\theta}, P_{T_X/\theta}$ be the respective distributions when learning probabilistic logic rules. Now let us define the semantics of a distribution $P_{T/\theta}$ in our case: $P_{T/\theta} = P(\pi(T/\theta))$, that is, distribution over the plan strings, which can be considered as lower semi-computable probability based on coding theory. We know,

$$\sum_{j \geq 1} 2^{-K(j)-c_j} P_j(y) \approx F(y|x) \tag{5}$$

where $F(y)$ is the cumulative distribution. So, NID $\delta(Y,Z)$ now becomes, $\delta(Y,Z) = \frac{max\left(K(y|z),K(z|y)\right)}{max\left(K(y),K(z)\right)}$ We know that $max\left(K(y),K(z)\right)$ is a normalizer. Thus, $\delta(Y,Z) < max\left(K(y|z),K(z|y)\right)$

$$\begin{aligned} max\left(K(y|z),K(z|y)\right) &= max\left(-logm(y|z),-logm(z|y)\right) \\ &= max\left(-log\frac{m(y,z)}{m(z)},-log\frac{m(y,z)}{m(y)}\right) \\ &= max\left(\left[-\log m(y,z)+\log m(z)\right], \right. \\ &\quad \left.\left[-\log m(y,z)+\log m(y)\right]\right) \end{aligned}$$

Under partial ordering max yields supremum

$$\begin{aligned} &\approx \mathbf{sup}\left|\log m(y) - \log m(z)\right| \\ &\approx \mathbf{sup}\left|\log F(y) - \log F(z)\right| \\ &\approx \mathbf{sup}\left|F(y) - F(z)\right| \log \text{ is monotonic} \end{aligned}$$

Significance of Propositions 1 and 2: Proposition 1 outlines how our proposed NCD-based metric is a valid conceptual distance. It is well understood that the true measure of conceptual distance is not straightforward and is subject to the semantic interpretation of the domain itself. But designing a unique distance metric based on the semantics of every domain limits the generality of any learning system. So NCD acts as a surrogate "conceptual distance." It is based on the notion that "if two concepts are fundamentally same the complexity of optimal action plans to realize the concepts should also be fundamentally same." NCD (or NID) essentially measures the difference in generative complexities of two plans. Also note that other types of distances that are limited to a syntactic level such as edit distance (or literal distance) will fail to capture the similarity or diversity between concept representations since the same concept can be represented with more than one theories that may vary in one or more literals.

Proposition 2, on the other hand, proves that our proposed metric is not limited to our specific scenario. It positions our work in the context of known statistical distance metrics and establishes its credibility as a valid solution. It proves how in a nondeterministic setting, that is, probabilistic logic formulation, our proposed metric generalizes to Kolmogorov–Smirnov statistic.

3.4.2. PAC Learnability

PAC analysis of GOCI follows from GOLEM for function-free *horn* clause induction (Muggleton and Feng, 1990). Let initial hypothesis space be $\mathcal{H}_0$ and the final be $\mathcal{H}^*$ ($s.t. T^* \in \mathcal{H}^*$).

Proposition 3 (Sample Complexity)**.** *Following Valiant (1984) and Mooney (1994), with probability* $(1-\delta)$*, the sample complexity of inducing the optimal theory* T^* *is:*

$$n^* = \mathcal{O}\left(\frac{1}{\epsilon}\left[d^L j^i \ln((tfm)) + \ln(\frac{1}{\delta})\right]\right) \tag{6}$$

where ϵ *is the regret,* n^* *- sample complexity of* $\mathcal{H}^*$*, i is the maximum depth of a variable in a clause and & j is the maximum arity. m - number of distinct predicates, t is the number of terms, p is the place and d is the distance of the current revision from the last known consistent theory, and L is the upper bound on the number of refinement steps (iterations).*

Proof Sketch for Proposition 3: *In our learning setting, the learned theory will always have nonzero uncertainty.* To understand the properties, we build upon the PAC analysis for recursive *rlgg* (Relative Least General Generalization) approach for function-free *Horn* clause learning shown by Muggleton and Feng (1990) in GOELM. With some restrictions, it applies here as well. Let n: denote the sample size and $\mathcal{H}$: the hypothesis space. Our approach can be considered as an *rlgg* approach with refinement steps. Note that constraint predicates that refine the clauses are not part of $\mathcal{K}$.

To begin with, we are interested in regret bounds for the initially learned hypothesis by the ILP learner $\mathcal{H}_0$, before refinement. We know from Valiant (1984), that with probability $(1-\delta)$ the sample complexity n for $\mathcal{H}_0$ is,

$$n \geq \frac{1}{\epsilon}\left(\ln(|\mathcal{H}_0|) + \ln(\frac{1}{\delta})\right) \tag{7}$$

where ϵ is the regret. Now, our ILP learner induces *ij*-determinate clauses (Muggleton and Feng, 1990), where i is the maximum depth of the clause and j is the maximum arity. In our problem setting, it can be proven that $|\mathcal{H}_0| = \mathcal{O}((tpm)^{j^i})$, where m is the number of distinct predicates in the language. t is the number of terms, and p is the place (for details about place, refer Muggleton and Feng, 1990). Also note that if j & i is bounded, then $j^i \leq c$). Mooney (1994) shows that for theory refinement/revision, sample complexity is expressed as,

$$n^* = \mathcal{O}\left(\frac{1}{\epsilon}\left[d^k \ln\left(|\mathcal{H}_0| + d + m\right) + \ln(\frac{1}{\delta})\right]\right) \tag{8}$$

where distance d to be the number of single literal changes in a single refinement step and k is the number of refinement/revision iterations. In **Algorithm 1**, we observe that at each iteration $\ell \leq L$, updates are with respect to the preferred constraint predicates $\mathcal{U}_\mathcal{A} \in \mathbb{U}$. Thus, we know that $k = L$. Substituting $|\mathcal{H}_0| = (tfm)^{j^i}$ and $j^i = c$ constant)in Equation (8) and ignoring the additive terms $d + m$ since $(tfm)^{j^i} >> d + m$, we get,

$$n = \mathcal{O}\left(\frac{1}{\epsilon}\left[d^L c \ln(tpm) + \ln(\frac{1}{\delta})\right]\right) \tag{9}$$

Proposition 4 (Refinement Distance)**.** *d is upper bounded by the expected number of literals that can be constructed out of the library of constraint predicates with human advice $\mathbb{E}_{\sim\mathcal{A}}\left[|\mathbb{U}|\right]$ and lower bounded by the conceptual distance between theory learned at two consecutive iterations since we adopt a greedy approach. If $Pr_\mathcal{A}(\mathcal{U})$ denotes the probability of a constraint predicate being preferred, then $|D_\ell - D_{\ell-1}| \leq d \leq \sum_{i=1}^{2^{(|\mathbb{U}|-1)} \times {}^tP_q} Pr_\mathcal{A}(\mathcal{U}_i)$ where $2^{(|\mathbb{U}|-1)} \times {}^tP_q$ is the maximum possible number of constraint literals and q is the maximum arity of the constraints. In case of only pairwise constraints, q =2.*

Proof Sketch for Proposition 4: The proof is straightforward and hence we present it in brief. In our setting to show that,

$$|D_\ell - D_{\ell-1}| \leq d \leq \sum_{i=1}^{2^{(|\mathbb{U}|-1)} \times {}^tP_q} P_\mathcal{A}(\mathcal{U}_i) \tag{10}$$

(where $2^{(|\mathbb{U}|-1)} \times {}^tP_q$ is the maximum number of constraint literals possible, since $\mathbb{U}$ is the library of constraint predicates) consider that the number of constraint predicates that can be picked up at any iteration is $2^{(|\mathbb{U}|-1)}$. To form constraint literals, we need to tie arguments to existing logical variables in the current theory. We have defined t to be the number of terms in the existing theory. Let q be the max arity of a constraint, thus total possible number of constraint literals are $2^{(|\mathbb{U}|-1)} \times {}^tP_q$. So if the distribution induced on the constraint literals by human advice $\mathcal{A}$ be $P_\mathcal{A}$, then $\sum_{i=1}^{2^{(|\mathbb{U}|-1)} \times {}^tP_q} P_\mathcal{A}(\mathcal{U}_i)$ is the expected number of literals added given the advice. Now this is the upper bound of d. Again d should at least be the conceptual distance between the new theory after constraint addition and the last consistent theory. Note d and conceptual distance D is not the same. Thus, it is the difference between the NCD of last theory to original example and current updated theory to the original example $|D_\ell - D_{\ell-1}|$.

Observe that if at each layer $\ell \leq L$ we add constraint predicates $\mathcal{U}_\ell$, then at layer ℓ, $d = \left|\{\mathcal{U}\}^\ell\right| \leq 2^{mt}P_q$ (assuming q is maximum arity of the constraint predicates). Also, as per our greedy refinement framework, at each layer ℓ, distance new theory $\mathbb{T}_\ell$ should at least be the change in conceptual distance.

Significance of Propositions 3 and 4: Propositions 3 and 4 aim to illustrate what the general sample complexity would be for a theory refinement-based RLGG clause learner such as GOCI and how the conceptual distance controls the complexity by establishing bounds on the refinement distance. Furthermore, the complexity is also subject to the maximum refinement iterations, which in turn is affected by human guidance. Thus, we establish the theoretical connection between the two dimensions of the contribution of this work.

Proposition 5 (Advice Complexity)**.** *From Equations (6) and (8), at convergence $\ell = L$, we get $\frac{n^* - |X|}{L}$ examples, on an average, for a concept $\mathcal{C}$ to be PAC learnable using* GOCI.

The proof is quite straightforward and hence we just discuss the brief idea behind it. Our input is sparse (one or few instances). GOCI elicits advice over constraints to acquire additional information. Let $|X|$ be the number of input examples. We query the human once at each layer and hence the maximum query budget is L. Given that the sample size is $|X|$, each query to the human must acquire information about at least $\frac{n^* - |X|}{L}$ examples, on an average, for our a concept $\mathcal{C}$ to be PAC learnable using our approach.

4. EVALUATION

We next aim to answer the following questions explicitly:

(Q1) Is GOCI effective in "one-shot" concept induction?
(Q2) How sample efficient is GOCI compared to baselines?
(Q3) What is the relative contribution of the novel scoring function versus human guidance toward performance?

Our framework extends a Java version of Aleph (Srinivasan, 2007). We modified the scoring function with NCD penalty computed via a customized SHOP2 planner (Nau et al., 2003). We added constraint sampling and human guidance elicitation iteratively (**Algorithm 1**).

4.1. Experimental Design

We compare GOCI with Aleph with no enhancements. We focus on the specific task of "one-shot concept induction," with a single input example for each of the several types of concepts and report aggregated precision. We consider precision because preference queries are meant to eliminate false positives in our case. To demonstrate general robustness of GOCI, beyond one-shot case, we experimented with varying sample sizes for each concept type and show learning curves for the same. We perform an ablation study to show the relative contribution of two important components of

GOCI: (a) novel scoring metric and (b) human guidance, that is, we compare against two more baselines (*ILP+Score* and *ILP+Guidance*). For every domain, we consider 10 different types of concepts (10 targets) and aggregate results over 5 runs.

Note that human guidance was obtained from distinct human experts for every run. The expertise level of all the advice providers was reasonably at par since they were chosen from the same pool of candidates with zero visibility and knowledge of our proposed framework. However, for all the human advice providers we assumed a basic level of knowledge in geometry or fundamentals of logic and reasoning. Additionally, we also explained each of the experimental domains to the human participants to create a similar level of awareness about the domains among all of them.

4.1.1. Domains

We employ four domains with varying complexity. Note that we have selected the below domains based on multiple considerations. The domain encoding need to be such that target concepts can be learned in a modular fashion (i.e., decomposable). Thus, the first two domains are structure construction domains either spatial (Minecraft) or chemical/molecular (CheBI). Spatial structures are implicitly modular (such as the $\mathbb{L}$-structure in **Figure 1**). Chemical entities, molecules/compounds/complexes, are similarly modular as well. The last two domains are fundamentally planning domains. However, they are also compositional in nature, that is, any planning goal is a composition task. For instance, machine structure in "Assembly" domain and cocktails, etc., in "Barman" domain. So these two domains do not just demonstrate learning modular/decomposable concepts but they also illustrate the plan induction feature of GOCI.

1. ***Minecraft (spatial structures):*** The goal is to learn discrete spatial concepts in a customized (Narayan-Chen et al., 2019) Project Malmo platform for Minecraft. Dialogue data in Malmo are available online, and we converted them into a logical representation. All structures are in terms of discrete atomic unit blocks (cubes). **Figure 5** shows examples of some spatial structures that GOCI was able to learn.
2. ***Chemical Entities of Biological Interest (ChEBI):*** ChEBI (Degtyarenko et al., 2007) is a compound database containing important structural features and activity-based information, for classification of chemicals, such as (1) molecular structure and (2) biological role. We model the Benzene molecule prediction task as molecular-compositional concepts. The data have predicates such as `SingleBond`, `DoubleBond`, and `HasAtom`.
3. ***Assembly (planning domain):*** Assembly is a planning domain, where different mechanical structure concepts are compositions of different parts and resources. Input is a conjunction of ground literals indicating ground plan demonstration (assuming total ordering).
4. ***Barman (planning domain):*** A standard planning domain where a bartender is supposed to follow certain recipes and sequence of techniques to create cocktails. The different cocktails are decomposable concepts in this setting.

4.2. Experimental Results

[Effective One-shot (Q1)] **Table 1** shows the performance of GOCI on one-shot concept learning tasks as compared to standard ILP. GOCI significantly outperforms ILP across all domains answering (Q1) affirmatively. Also, note that GOCI is very "query" efficient as observed from the small average

TABLE 1 | Results for one-shot concept learning.

Domain	Approach	Avg. precision	#Queries
Minecraft	Goci	0.85	5.5 ± 3
	ILP	0.35	–
Assembly	Goci	0.65	16.5 ± 4
	ILP	0.2	–
ChEBI	Goci	0.615	13.1 ± 2.13
	ILP	0.45	–
Barman	Goci	0.7	10.5 ± 5.4
	ILP	0.51	–

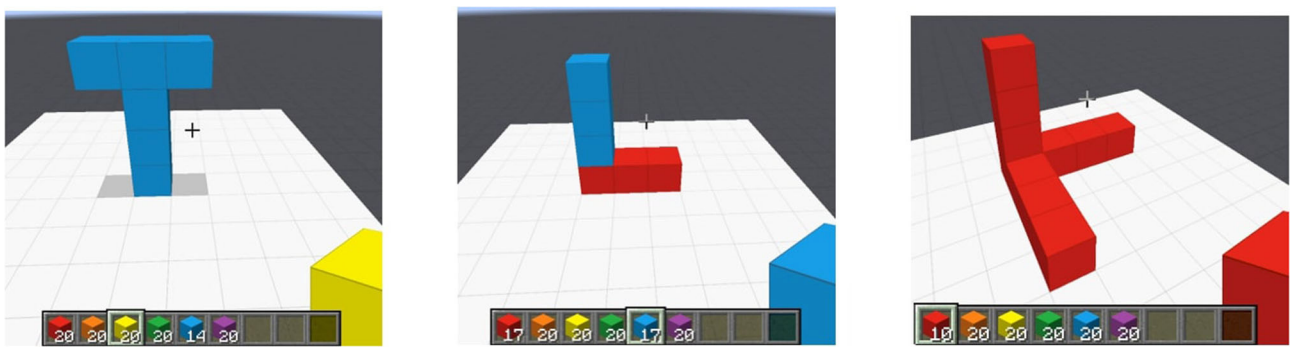

FIGURE 5 | Instances of spatial concepts in Minecraft. **(Left)** Upright Tee, **(Middle)** Upright L, **(Right)** *Orthogonal overlapping Ls*.

number of queries posed in the case of each domain. Note that in the case of CheBI, the number of queries is the highest among all the domains. This can be attributed to that fact that CheBI is a domain, which requires a certain degree of understanding of fundamental chemistry (chemical bonds and their types, molecules, atoms, etc.). Thus, some of the human participants required more iterations (consequently more queries) to converge to the most relevant set of constraint literals, given the difference of their prior understanding of school chemistry.

Query efficiency is an important consideration in any learning paradigm that leverages human guidance, since controlling the cognitive load on the human expert is critical. So, in general, the observed average query numbers being reasonably low across all domains corroborates our theoretical advice complexity (section 3.4.2).

[Sample Efficiency (Q2)] In **Figure 6**, we observe that GOCI converges within significantly smaller sample size across all domains, thus, supporting our theoretical claims in section 3.4. In ChEBI, though, quality of planner encoding might explain mildly lower precision yet GOCI does perform significantly better than vanilla ILP learner. In ChEBI, we see that the sample efficiency is not vastly distinct. One of the possible reasons could be the sub-optimal encoding of the planning domain language, which is necessary for NCD computation, for this task. If we can improve the planner setup for this domain, then we will likely be able to observe enhanced performance.

[Relative contribution (Q3)] **Figure 7** validates our intuition that both components (scoring function and human-guidance) together make GOCI a robust one-shot (sample-efficient) concept induction framework. Though human guidance, alone, is able to enhance the performance of a vanilla ILP learner in sparse samples, yet it is not sufficient for optimal performance. In contrast, although the advantage of our novel distance-penalized scoring metric is marginal in sparse samples, it is essential for optimal performance at convergence.

5. DISCUSSION

The most important conclusion from the experiments is that when available, the guidance along with the novel score leads to a jump-start, better slope and in some cases, asymptotically sample

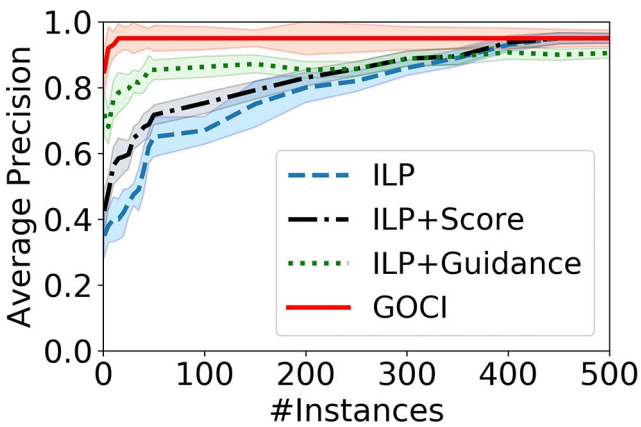

FIGURE 7 | Results of ablation study on Minecraft domain. Relative contribution of our distance-penalized score vs. human guidance.

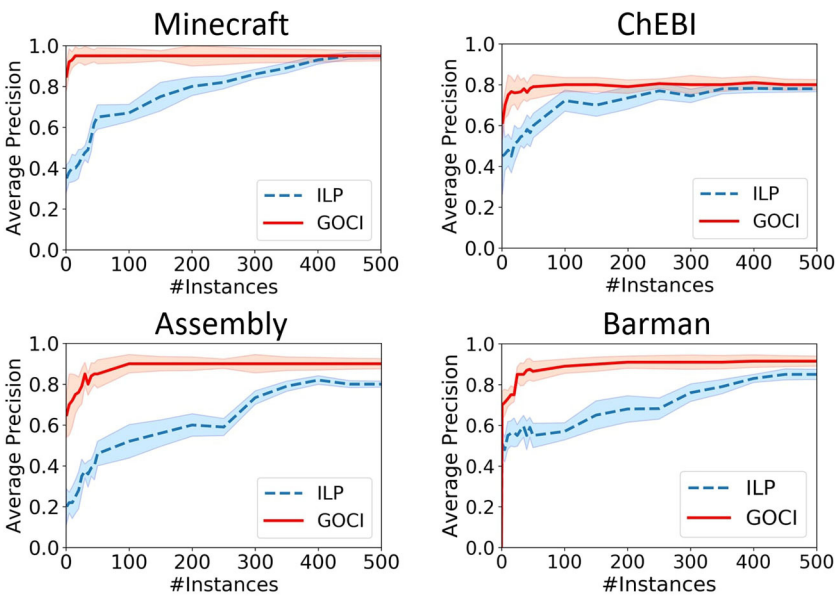

FIGURE 6 | Learning curves for varying sample size to compare the sample efficiency of Guided One-shot Concept Induction (GOCI) and inductive logic program (ILP). Top two plots are with respect to structural composition domains-Minecraft & ChEBI and the bottom two are for planning domains: Assembly and Barman (best viewed in color).

efficient with a fraction of the number of instances needed than merely learning from data.

Another important aspect to note here is that our experimental setup did not attempt to ensure in any way that the quality of guidance provided by the human participants is optimal. The formulation of the objective function, itself, in GOCI is designed to handle sub-optimal human advice implicitly in a seamless manner. The two primary features in the design that make GOCI robust to advice quality are as follows:

1. As explained earlier and shown in Equation (3), human advice and conceptual distance deal with two distinct aspects of the search process. Human advice controls the size and nature of the search space while conceptual distance ensures the quality of the candidates. Advice and distance have a balancing effect on each other, and thus, it is our novel conceptual distance that makes GOCI robust to bad advice.
2. Also, the nature of human advice in our setting is of choosing the most useful set of "constraint predicates" among the set of candidate constraints. Now the candidates are generated by GOCI in a conservative fashion selecting only the ones that are logically valid for the theory learned at the current iteration of revision. Thus, human experts have very little option of choosing an invalid or extremely unlikely constraint predicate.

Our ablation study in **Figure 7** also supports our analysis. On closer inspection, we see that it is due to our novel distance penalized scoring function (ILP+Score) that ensures convergence to an optimal solution. Human advice (ILP+Guidance) contributes to sample efficiency.

6. CONCLUSIONS

We developed a human-in-the-loop one-shot concept learning framework in which the agent learns a generalized representation of a concept as FOL rules, from a single (few) positive example(s). We make two specific contributions: deriving a new distance measure between concepts and allowing for richer human inputs than mere labels, solicited actively by the agent. Our theoretical and experimental analyses show the promise of GOCI method. An exhaustive evaluation involving richer human inputs including varying levels of expertise and analyzing our claim that learning performance of GOCI is robust to expertise levels (which should only affect query efficiency) is an immediate future research objective. Integration with hierarchy learning also remains an interesting direction for future research.

DATA AVAILABILITY STATEMENT

All datasets generated for this study are included in the article/supplementary material.

AUTHOR CONTRIBUTIONS

MD and SN contributed equally to the ideation. MD and NR led the empirical evaluation. MD, NR, SN, and JD contributed nearly equally to the manuscript preparation. All authors contributed to the article and approved the submitted version.

FUNDING

MD and SN gratefully acknowledge the support of CwC Program Contract W911NF-15- 1-0461 with the US Defense Advanced Research Projects Agency (DARPA) and the Army Research Office (ARO). SN and NR gratefully acknowledge AFOSR award FA9550-18-1-0462. Any opinions, findings, and conclusion or recommendations are those of the authors and do not necessarily reflect the view of the DARPA, ARO, AFOSR, or the US government.

ACKNOWLEDGMENTS

The authors acknowledge the support of members of STARLING lab for the discussions.

This article was released as a preprint to arXiv as part of the StarAI 2020 workshop (Das et al., 2020).

REFERENCES

Bart, E., and Ullman, S. (2005). "Single-example learning of novel classes using representation by similarity," in *BMVC* (Oxford).

Braziunas, D., and Boutilier, C. (2006). "Preference elicitation and generalized additive utility," in *AAAI* (Boston, MA).

Chick, H. L. (2007). "Teaching and learning by example," in *Mathematics: Essential Research, Essential Practice* (Hobart, TAS).

Das, M., Odom, P., Islam, M. R., Doppa, J. R., Roth, D., and Natarajan, S. (2018). "Preference-guided planning: an active elicitation approach," in *AAMAS* (Stockholm).

Das, M., Ramanan, N., Doppa, J. R., and Natarajan, S. (2020). "One-shot induction of generalized logical concepts via human guidance," in *StarAI-20 Workshop*. Available online at: https://arxiv.org/abs/1912.07060

Degtyarenko, K., De Matos, P., Ennis, M., Hastings, J., Zbinden, M., McNaught, A., et al. (2007). Chebi: a database and ontology for chemical entities of biological interest. *Nucleic Acids Res.* 36, D344–D350. doi: 10.1093/nar/gkm791

Eiben, A. E., and Smith, J. E. (2015). *Interactive Evolutionary Algorithms*. Berlin; Heidelberg: Springer Berlin Heidelberg, 215–222.

Flener, P. (1997). "Inductive logic program synthesis with dialogs," in *ILP* (Prague).

Friend, M., Khaled, M., Lefever, K., and Székely, G. (2018). Distances between formal theories. Available online at: http://philsci-archive.pitt.edu/14849/

Fung, G. M., Mangasarian, O. L., and Shavlik, J. W. (2003). "Knowledge-based support vector machine classifiers," in *NIPS* (Vancouver, BC).

Goldman, R. P., and Kuter, U. (2015). "Measuring plan diversity: pathologies in existing approaches and a new plan distance metric," in *AAAI* (Austin, TX).

Horn, A. (1951). On sentences which are true of direct unions of algebras. *J. Symbol. Logic* 16, 14–21.

Kitzelmann, E., and Schmid, U. (2006). Inductive synthesis of functional programs: an explanation based generalization approach. *J. Mach. Learn. Res.* 7, 429–454.

Kozerawski, J., and Turk, M. (2018). "Clear: cumulative learning for one-shot one-class image recognition," in *CVPR* (Salt Lake City, UT).

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science* 350, 1332-1338. doi: 10.1126/science.aab3050

Mitchell, T. (1997). *Machine Learning*. McGraw-Hill Higher Education.

Mooney, R. J. (1994). A preliminary PAC analysis of theory revision, *Comput. Learn. Theory Nat. Learn. Syst.* 3, 43–53.

Muggleton, S. (1988). "Machine invention of first-order predicates by inverting resolution," in *Proc. of 5th Machine Learning Conference* (Ann Arbor, MI).
Muggleton, S. (1991). Inductive Logic Programming. *N. Gen. Comput*. 8, 295–318.
Muggleton, S. (1995). Inverse Entailment and Progol. *N. Gen. Comput*. 13, 245–286.
Muggleton, S., and Feng, C. (1990). "Efficient induction of logic programs," in *ALT* (Tokyo).
Narayan-Chen, A., Jayannavar, P., and Hockenmaier, J. (2019). "Collaborative dialogue in Minecraft," in *ACL* (Florence).
Nau, D., Au, T.-C., Ilghami, O., Kuter, U., Murdock, J. W., Wu, D., et al. (2003). Shop2: An HTN planning system. *J. Artif. Intell. Res*. 20, 379–404. doi: 10.1613/jair.1141
Odom, P., Khot, T., Porter, R., and Natarajan, S. (2015). "Knowledge-based probabilistic logic learning," in *AAAI* (Austin, TX).
Pazzani, M. J. (1992). *An Information-Based Approach to Integrating Empirical and Explanation-Based Learning*. Tokyo: Inductive Logic Programming.
Preiss, K., and Shai, O. (1989). Process planning by logic programming. *Robot. Comput. Integr. Manufact*. 5, 1–10. doi: 10.1016/0736-5845(89)90025-2
Quinlan, J. R. (1990). Learning logical definitions from relations. *Mach. Learn*. 5, 239–266.
Rouveirol, C. (1992). *Extensions of Inversion of Resolution Applied to Theory Completion*. Tokyo: Inductive Logic Programming.
Sammut, C., and Banerji, R. B. (1986). "Learning concepts by asking questions," in *Machine Learning: An Artificial Intelligence Approach, Vol. 2*, eds R. S. Michalski, J. G. Carbonell, and T. M. Mitchell (Morgan Kaufmann), 167–192.
Shavlik, J. W., and Towell, G. G. (1989). "Combining explanation-based learning and artificial neural networks," in *ICML*.
Srinivasan, A. (2007). *Aleph: A Learning Engine for Proposing Hypotheses*. Available online at: http://web2.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/aleph.pl
Stroulia, E., and Goel, A. K. (1994). "Learning problem-solving concepts by reflecting on problem solving," in *ECML* (Catania).
Tax, D. M. J. (2001). *One-class classification: concept learning in the absence of counter-examples* (Ph.D. thesis). TU Delft, Delft, Netherlands.
Valiant, L. G. (1984). A theory of the learnable. *Commun. ACM*. 27, 1134–1142.
Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE Trans. Neural Netw*. 10, 988–999,
Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016). "Matching networks for one shot learning," in *NIPS* (Barcelona).
Vitányi, P. M. (2013). Conditional kolmogorov complexity and universal probability. *Theor. Comput. Sci*. 501, 93–100. doi: 10.1016/j.tcs.2013.07.009
Vitányi, P. M., Balbach, F. J., Cilibrasi, R. L., and Li, M. (2009). "Normalized information distance," in *Information Theory and Statistical Learning* (Springer), 45–82.
Wang, Y.-X., Girshick, R., Hebert, M., and Hariharan, B. (2018). "Low-shot learning from imaginary data," in *CVPR* (Salt Lake City, UT).
Wendt, K., Cortés, A., and Margalef, T. (2010). Knowledge-guided genetic algorithm for input parameter optimisation in environmental modelling. *Proc. Comput. Sci*. 1, 1367–1375. doi: 10.1016/j.procs.2010.04.152
Wilson, R. H., and Latombe, J.-C. (1994). Geometric reasoning about mechanical assembly. *Artif. Intell*. 71, 371–396. doi: 10.1016/0004-3702(94)90048-5
Wong, M. L., and Leung, K. (1997). Evolutionary program induction directed by logic grammars. *Evol. Comput*. 5, 143–180.

Conflict of Interest: MD was employed by the company Samsung R&D Institute India - Bangalore.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.