Towards Unified Representations of Knowledge Graph and Expert Rules for Machine Learning and Reasoning

Zhepei Wei[1,3], **Yue Wang**[2], **Jinnan Li**[3,5], **Zhining Liu**[1,3], **Erxin Yu**[1,3], **Yuan Tian**[1,3], **Xin Wang**[1,3], **Yi Chang**[1,3,4*]

[1]School of Artificial Intelligence, Jilin University
[2]School of Information and Library Science, University of North Carolina at Chapel Hill
[3]Key Laboratory of Symbolic Computation and Knowledge Engineering, Jilin University
[4]International Center of Future Science, Jilin University
[5]College of Computer Science and Technology, Jilin University
{weizp19, jnli21, znliu19, yuex19}@mails.jlu.edu.cn, wangyue@email.unc.edu,
{yuantian, xinwang, yichang}@jlu.edu.cn

Abstract

With a knowledge graph and a set of if-then rules, can we reason about the conclusions given a set of observations? In this work, we formalize this question as the *cognitive inference* problem, and introduce the Cognitive Knowledge Graph (CogKG) that unifies two representations of heterogeneous symbolic knowledge: expert rules and relational facts. We propose a general framework in which the unified knowledge representations can perform both learning and reasoning. Specifically, we implement the above framework in two settings, depending on the availability of labeled data. When no labeled data are available for training, the framework can directly utilize symbolic knowledge as the decision basis and perform reasoning. When labeled data become available, the framework casts symbolic knowledge as a trainable neural architecture and optimizes the connection weights among neurons through gradient descent. Empirical study on two clinical diagnosis benchmarks demonstrates the superiority of the proposed method over time-tested knowledge-driven and data-driven methods, showing the great potential of the proposed method in unifying heterogeneous symbolic knowledge, i.e., expert rules and relational facts, as the substrate of machine learning and reasoning models. The source code and data are released online[1].

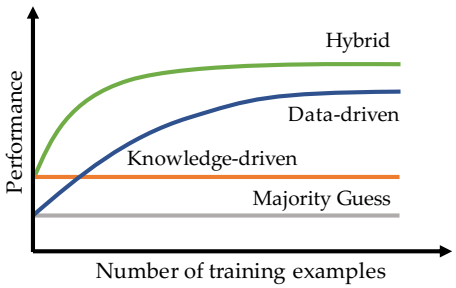


Figure 1: Illustration of impacts of training examples on different reasoning paradigms (with fixed prior knowledge). Note that the curves start w/o pre-training.

1 Introduction

Symbolic reasoning methods such as rule-based expert systems (Buchanan and Shortliffe, 1984) are reliable and interpretable in solving complex inference problems in specialized domains, but are also difficult to generalize because eliciting a comprehensive set of rules from human experts is costly and time-consuming. Recently, knowledge graph (KG) as a flexible representation of symbolic knowledge has been proven successful for knowledge-based reasoning (Bordes et al., 2013) by utilizing the distributed representations to generalize from known facts to unseen yet probably true facts, which is also known as the knowledge graph completion task (Lin et al., 2015). However, such models can only represent and reason about multi-relational data in the form of (*subject*, *predicate*, *object*) triples (Liben-Nowell and Kleinberg, 2003), not conditional *if-then* rules. Therefore, current knowledge graph embedding models are not suited to solve inference problems where conclusions (outcomes) can be inferred from a set of observations.

This current work is motivated by one overarching question: can we unify the representation of above heterogeneous symbolic knowledge to perform complex inference tasks? More concretely, we study the following research question. With a large-scale KG with rich relational facts and a moderate set of if-then rules as the prior knowledge, can we reason about the most likely conclusion(s) given a set of observations? With the rapid development of knowledge graph, the *knowledge acquisition bottleneck* (Muggleton and De Raedt, 1994) is greatly alleviated, making it much more practicable to jointly utilize the knowledge and data

*Corresponding Author
[1]http://github.com/jinnanli/CogKG

for learning systems in today than in the past. Recent studies have shown great success in integrating the knowledge into data-driven models, and such hybrid learning system normally achieves more favorable performance than traditional methods, as presented in Fig. 1. However, there is a general absence of sufficient labeled data in some high-stake scenarios such as medical diagnosis. Moreover, such critical domains' inherent nature strictly mandates the models to be trustworthy and interpretable. These high-demanding characteristics directly challenge existing vulnerable knowledge-driven methods and data-hungry machine learning methods, and the solution still remains underexplored[2] (von Rueden et al., 2021).

In this work, we formalize the above challenge as the *cognitive inference* problem and introduce three design goals for the model to address this problem: 1) The ability to extensively inherit existing symbolic knowledge. The model is expected to leverage not only if-then rules, but also large number of facts in knowledge graphs. 2) The ability to directly utilize existing symbolic knowledge in the reasoning procedure. This allows the model to make decent predictions based on prior knowledge, even when it is not trained. Moreover, it makes the model's reasoning process interpretable. 3) The ability to be continuously optimized when training data is available. This enables the model to improve like any machine learning models. More importantly, it ensures the model's robustness so that it adapts to the nuances of real-world data that are not encoded in prior symbolic knowledge.

To achieve the above goals, we first introduce the cognitive knowledge graph (CogKG), which represents relational facts and expert rules in a unified framework. Specifically, it is a directed hypergraph with entities as nodes, and the relations or expert rules as edges. Then, we propose a novel inference framework called COGINFER that bridges the knowledge-driven and data-driven reasoning paradigms, which not only utilize explicit knowledge representations but also harvest knowledge from training examples if applicable. More precisely, it performs reasoning with symbolic knowledge, and the reasoning process could be further optimized with labeled data towards better performance. In this way, we aim to combine the symbolic reasoning and statistical learning in the same general framework COGINFER, which make our method achieve the design goals as stated above and stand out from existing works.

To make fair comparisons with existing knowledge-driven and data-driven baselines, we investigate the cognitive inference problem under both unsupervised and supervised settings. Extensive experiments on two clinical diagnosis benchmarks show that the COGINFER successfully learns from both symbolic knowledge and labeled data to address the proposed new inference task, substantially surpassing strong data-driven baselines. Even without any training examples, it still outperforms existing knowledge-driven baselines that only harvests either expert rules or knowledge graph, demonstrating the great potential of the proposed framework. The main contributions of this work are three-fold:

- We introduce a novel cognitive inference problem that reasons about conclusions from observations, which directly challenges existing methods.
- In light of this challenge, we first introduce the cognitive knowledge graph (CogKG) that represents expert rules and relational facts in a unified manner, and then develop a general framework that bridges the knowledge-driven and data-driven reasoning paradigm.
- Extensive experiments demonstrate the effectiveness of the proposed method in utilizing unified symbolic knowledge and labeled data for machine learning and reasoning.

2 The Cognitive Inference Problem

2.1 Problem Formulation

We first introduce our notations. A **knowledge graph** (KG) consists of relational facts $\mathcal{F} = \{(s_i, p_i, o_i)\}_{i=1}^{N}$, where (s_i, p_i, o_i) is a relational triple consisting of subject entity s_i, predicate p_i, and object entity o_i. The vertex set of the KG is $V = \cup_{i=1}^{N}\{s_i, o_i\}$ and its edge set is $E^e = \cup_{i=1}^{N}\{p_i\}$. The collection of **expert rules** is denoted as $\mathcal{R} = \{A_i \xrightarrow{r_i} B_i\}_{i=1}^{M}$, where $A_i \xrightarrow{r_i} B_i$ is a rule that expresses "if A_i are observed then B_i are true". $A_i, B_i \subset V$ are small sets of entities and r_i is a *hyperedge* that connects two *sets* of entities. The hyperedge set is denoted as $E^r = \cup_{i=1}^{M}\{r_i\}$. **Labeled data** $\mathcal{L} = \{(Q_i, C_i)\}_{i=1}^{L}$ is a collection of query-conclusion pairs. $Q_i, C_i \subset V$ are small sets of entities. In machine learning terms, the query

[2] See detailed discussion in Appendix A.

Table 1: Important notations and descriptions.

Notations	Descriptions
$\mathcal{F}$	Relational facts
$\mathcal{R}$	Expert rules
$\mathcal{L}$	Labeled data
$\mathcal{G}$	CogKG, $\mathcal{G} = (V, E)$
V	Entities
E	Edges, $E = \{E^e, E^r\}$
E^e	Relation edges
E^r	Rule hyperedges
Q	A query, a small set of entities
C	A conclusion, a small set of entities
$\mathcal{G}^Q$	InferGraph of query Q, $\mathcal{G}^Q \subset \mathcal{G}$
$\mathcal{E}$	Distributed representations of relational facts
P	Rule-generated neuron matrix
U	KG-generated neuron matrix
X	Final input neuron matrix
W	Weight matrix before output neurons

Q_i are input features and the conclusion C_i are prediction targets. In this work, we instead use "query" and "conclusion" to emphasize the inference nature of our problem.

The ***cognitive inference problem*** is to infer the conclusion $C \in V$ for a given query $Q \in V$. The inference is *unsupervised* if it only makes use of the knowledge graph $\mathcal{F}$ and expert rules $\mathcal{R}$; it is *supervised* if it also makes use of the label data $\mathcal{L}$.

2.2 Task Preliminaries

As the cognitive inference problem involves utilizing different resources for learning and reasoning, we assume each of them has been properly prepared before the task begins, as detailed below.

(1) **Knowledge Graph.** We assume access to a large-scale knowledge graph relevant to the problem domain. It is typically represented in the form of *(subject, predicate, object)* triples (Ji et al., 2021). These relational facts can be manually collected or automatically extracted from texts though natural language processing technologies such as named entity recognition (Yadav and Bethard, 2018; Yang et al., 2020) and relation classification (Yu et al., 2020; Han et al., 2020).

(2) **Expert Rules.** We assume access to a set of if-then rules encoding the expert knowledge of the problem domain. They are conditional statements which posit that a conclusion is true if the premises are satisfied by the input observations. It can be elicited from experts with domain knowledge. It can also be learned from domain data via machine learning and data mining (e.g., structure learning (Khosravi et al., 2010), decision tree (Quinlan, 1987), association rule mining (Han et al., 2000)).

(3) **Labeled Data.** Labeled data contains instances of queries and their corresponding conclusions in the problem domain. In this work, we consider the domain of medical diagnosis. Each piece of labeled data is a diagnosis record, where a query is a set of observed symptoms and a conclusion is a diagnosed disease. Labeled data are used for training (in supervised setting) and evaluation (in both supervised and unsupervised settings).

(4) **Entity Alignment.** The above resources may use different surface forms to refer to the same entity. It is crucial to align different surface forms using the same entity in the KG. This procedure can be done manually or assisted with entity disambiguation tools (Dredze et al., 2010).

3 Proposed Methods

3.1 Cognitive Knowledge Graph

To solve the above cognitive inference problem, we first introduce the **Cog**nitive **K**nowledge **G**raph (**CogKG**), which unifies the representation of relational facts and expert rules, and then develop a general reasoning framework based on it. As presented in Fig. 2, the CogKG is a directed hypergraph with entities as nodes, and the relations or rules as edges. In this case, the relation edge connects two entities and then forms a relational fact. In contrast, the rule hyperedge connects two sets of entities and then form a expert rule. We denote the cognitive knowledge graph as $\mathcal{G} = (V, E)$, where V is the entity set and $E = \{E^e, E^r\}$. In particular, the relation edges and rule hyperedges are E^e and E^r, respectively. The important notations and descriptions are in Table 1.

3.2 The General CogInfer Framework

With rich cognitive knowledge of expert rules and relational facts, we propose CogInfer, a general framework performing machine reasoning based on the CogKG. As presented in Alg. 1, the reasoning procedure for the cognitive inference problem includes three steps. Firstly, we perform knowledge representation learning on the relational facts of CogKG $\mathcal{G}$ and obtain the distributed representations of involved nodes and relational edges, i.e., $\mathcal{E} = \mathbf{RepreLearn}(V, E^e)$.[3] Secondly, a task-specific InferGraph $\mathcal{G}^Q$ is constructed from $\mathcal{G}$, which identifies the inference space for query

[3]This can be done by any knowledge graph embedding methods (Ji et al., 2021). Here we adopt the widely used TransE (Bordes et al., 2013) as a typical technique.

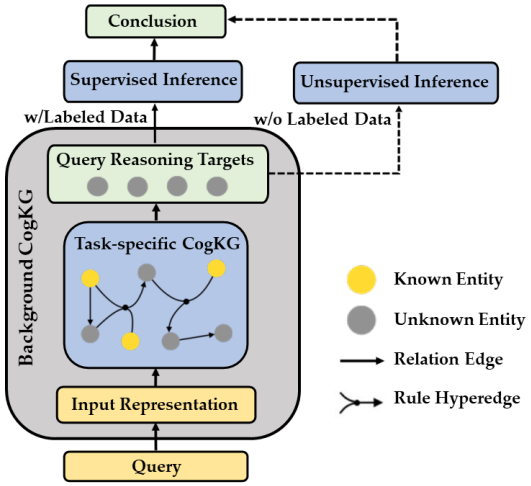


Figure 2: The general COGINFER framework. The query and conclusion are both aligned to CogKG.

Algorithm 1 COGINFER

Require: $\mathcal{G} = (V, E)$, $Q = \{v_1, ..., v_i..., v_L | v_i \in V\}$

Learn embeddings $\mathcal{E}$ for nodes and edges of $\mathcal{G}$
Create $\mathcal{G}^Q$ from $\mathcal{G}$ w.r.t. query Q ▷ Alg. 2
Perform inference based on $\mathcal{G}^Q$ and $\mathcal{E}$ ▷ Sec. 3.3 / Sec. 3.4
return conclusion C

Algorithm 2 InferGraph Construction

Require: $\mathcal{G}, Q$

Initialization: add entities in Q as nodes to $\mathcal{G}^Q$, $E^r_{mem} = \emptyset$
assign V_{cur} with nodes in $\mathcal{G}^Q$
E^r_{cur} = **GetLinkedRuleEdges**$(\mathcal{G}, V_{cur})$
while $E^r_{cur} - E^r_{mem}$ is not empty **do**
  for each $e^r_i \in E^r_{cur} - E^r_{mem}$ **do**
    V_i = **GetLinkedEntNodes**$(\mathcal{G}, e^r_i)$
    add rule e^r_i and nodes V_i to $\mathcal{G}^Q$
  expand E^r_{mem} with E^r_{cur}
  assign V_{cur} with nodes in $\mathcal{G}^Q$
  E^r_{cur} = **GetLinkedRuleEdges**$(\mathcal{G}, V_{cur})$
return InferGraph $\mathcal{G}^Q$

Q. Lastly, the COGINFER checks every possible conclusion delivered in the inference space by reasoning with curated rules in $\mathcal{G}^Q$ and the distributed representations $\mathcal{E}$. Specifically, it performs *unsupervised inference* (Section 3.3) or *supervised inference* (Section 3.4) depending on the availability of labeled data. The general COGINFER framework is presented in Fig. 2.

For each query $Q = \{v_1, ..., v_i..., v_L | v_i \in V\}$,

Algorithm 3 Unsupervised Inference

Require: $\mathcal{G}^Q, Q, \mathcal{E}$

Initialization: add rules in $\mathcal{G}^Q$ to E^r_{mem}, $C = \emptyset$
assign V_{knw} with entities in Q
repeat
  assign V_{mem} with V_{knw}
  for each $e^r_i \in E^r_{mem}$ **do**
    for each v_u in premise **do**
      LinkPrediction$(v_u, V_{knw}, \mathcal{E})$
    expand C with **ApplyRule**(e^r_i, V_{knw})
    expand V_{knw} with C
until $V_{knw} - V_{mem}$ is empty
return C

we create a task-specific InferGraph $\mathcal{G}^Q$ by iteratively identifying the closure of the involved rules and connected entities from the task-free background CogKG $\mathcal{G}$. The construction of InferGraph is detailed in Alg. 2. When expanding the rules, we only consider those where the premise requires at least one registered entity of the closure.

Specifically, **GetLinkedRuleEdges**$(\mathcal{G}, V)$ returns a set of rules of CogKG $\mathcal{G}$ in which the entity set in premise overlaps with V. **GetLinkedEntNodes**$(\mathcal{G}, e^r)$ returns a set of entity nodes of $\mathcal{G}$ that are linked with the rule e^r, i.e, those entities in premise and conclusion of this rule. In other words, $\mathcal{G}^Q \subset \mathcal{G}$ is a small sub-graph of the background CogKG. The entity nodes in this graph are particularly categorized into two sets, namely, *Known Entity* set and *Unknown Entity* set, representing the status of certainty. We use **certainty factor** (CF) (Buchanan and Shortliffe, 1984) to manage the uncertainty of the nodes carried out in the ensuing reasoning steps. Specifically, a CF of 0 represents unknown. Positive and negative CFs represent True and False values respectively, with increasing confidence as the number approaches 1 or -1. In our case, this CF is used to indicate the confidence in the presence or absence of symptoms or diseases. The logical AND and OR operations on two CFs a, b are defined as follows:

$$\text{AND}(a, b) = \min(a, b)\ , \tag{1}$$

$$\text{OR}(a, b) = \begin{cases} a + b - ab, & \text{if } a, b \geq 0; \\ a + b + ab, & \text{if } a, b < 0; \\ \frac{a+b}{1-\min(|a|,|b|)}, & \text{otherwise.} \end{cases} \tag{2}$$

3.3 Unsupervised Inference

In unsupervised inference, we conduct reasoning over the InferGraph by applying expert rules or link prediction with the learned distributed representations, i.e., to deduce the CF of unknown entities given a set of rules and known entities. As presented in Alg. 3, for each rule in the InferGraph, we check if the *Known Entity* set satisfy the corresponding premises. If satisfied, then we can apply this rule and deduce a new non-zero CF for the unknown entity and remove it from the *Unknown Entity* set. Otherwise, we check if there is any unknown entity in the premise could be deduced through link prediction. Formally, **LinkPrediction** takes as input the concerned unknown entity v_u, current known entity set V_{knw}, the learned embeddings $\mathcal{E}$, and outputs the CF of v_u. Let $|E^e|$ denote the pre-defined predicate set in E^e. Each known entity $v_i \in V_{knw}$ along with v_u forms a candidate triple (v_i, p_j, v_u) under a certain predicate $p_j \in |E^e|$. The cosine similarity between $(\vec{v_i} + \vec{p_j})$ and $\vec{v_u}$ is used to represent the CF given by such triple[4]:

$$CF_{ij} = CosSim(\vec{v_i} + \vec{p_j}, \vec{v_u}) \tag{3}$$

Particularly, the CF will be reset to 0 if the calculated result does not surpasses a preset threshold. By applying the OR operation over CFs carried by all such triples, the CF of the target unknown entity v_u is determined. **ApplyRule** takes as input the concerned rule e^r and current known entity set V_{knw}. If the rule is applicable, i.e., the premise is satisfied by the V_{knw}, we will apply the AND operation over CFs of all premise entities followed by multiplication with CF of the rule itself to determine the CF of conclusion entity led by the rule. We repeat the above procedure and record every conclusion until no more entity could be deduced.

3.4 Supervised Inference

So far, we have presented how the CogInfer performs unsupervised machine reasoning with cognitive knowledge of relational facts and if-then rules while without any labeled training data. With the same architectural backbone, it can be easily extended to a trainable supervised model and collectively learn from the knowledge and labeled data. To keep the explainability of CogInfer, we implement it as a simple neural network with only one fully connected layer between the input and output neurons, as presented in Fig. 3.

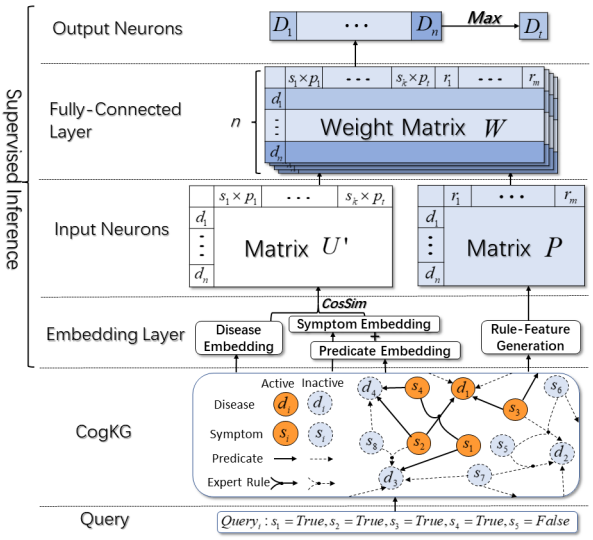


Figure 3: The trainable implementation of CogInfer.

In this section, we present how the cognitive knowledge of if-then rules and relational facts are utilized to generate explainable neurons[5] as part of the model and elaborate on the instantiation of the trainable implementation of CogInfer.

3.4.1 Rule-generated Neurons

In unsupervised inference, each applicable rule in the InferGraph gives a CF attached to a specific reasoning target. In other words, a single rule generates a target-specific scalar feature for each query and thus a rule set will give a collection of such scalar features. However, the number of reasoning targets are subject to the input query, leading to a unstable feature space as the query changes. In contrast, in supervised inference, the reasoning targets are fixed as the pre-defined set of labels. This inspires a macro perspective to consider all rules as a whole and treat the rule-generated CFs as inherently explainable neurons with respect to the input query. Formally, given m if-then rules, we defined the rule-generated neuron matrix P as follows.

$$P = \begin{array}{c} \begin{matrix} r_1 & r_2 & \cdots & r_m \end{matrix} \\ \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nm} \end{bmatrix} \end{array} \begin{matrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{matrix} \tag{4}$$

[4] In this work, we use arrowheaded letter to represent the corresponding vector in $\mathcal{E}$.

[5] Throughout this paper, the term *neurons* represent the feature units with explicit semantics produced by symbolic knowledge.

where p_{ij} is the CF given by r_j regarding reasoning target, i.e., disease d_i. For example, in our case, the input query x is a set of symptoms and we denote the corresponding vector as $\vec{x} \in \mathbb{R}^k$, where k is the dimension of symptom space. It comprises of a set of binary values (0/1), representing the presence of the corresponding symptom. Similarly, we can represent the rule vector in the same space as input query. Each element of $\vec{r}$ is also a binary value, indicating if the corresponding symptom is required in its premise. As a rule only has one predefined CF for a specific disease, we heuristically assign $p_{ij} = 0$ for all d_i that is not included in the conclusion of r_j or the rule itself is not applicable with input query.

$$p_{ij} = \alpha_j \times \mathbb{I}(\vec{x} \cdot \vec{r_j} == sum(\vec{r_j})) \tag{5}$$

where α_j is the CF of rule r_j, $\mathbb{I}$ is the indicator function that returns 1 if the condition is true and 0 otherwise. In this way, the original neurons of input query in the symptom space is extended to explainable rule-generated neurons in a more expressive space.

3.4.2 KG-generated Neurons

Likewise, the large amount of relational facts in KG can be utilized in the same manner. For each predicate $p_* \in |E^e|$, any dimension s_i in the original symptom space $\mathbb{R}^k$ together with reasoning target d_j forms a relational triple (s_i, p_*, d_j), which will lead to a CF attached to the reasoning target (as described in Section 3.3). Taking all dimensions into account, we can obtain a matrix of CFs under the specific predicate p_*. Therefore, the KG-generated neuron matrix U is defined as follows.

$$U = [U_{p_1}, ..., U_{p_i}, ..., U_{p_t}] \tag{6}$$

where t is the size of predefined predicate set $|E^e|$. Specifically, for each predicate p_*, the matrix U_{p_*} is defined as follows.

$$U_{p_*} = \begin{array}{c} \begin{matrix} s_1 & s_2 & \cdots & s_k \end{matrix} \\ \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1k} \\ u_{21} & u_{22} & \cdots & u_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ u_{n1} & u_{n2} & \cdots & u_{nk} \end{bmatrix} \end{array} \begin{matrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{matrix} \tag{7}$$

where u_{ij} represents the CF given by the relational fact (s_i, p_*, d_j).

$$u_{ij} = cos(\theta) = \frac{(\vec{s_i} + \vec{p_*}) \cdot \vec{d_j}}{\|(\vec{s_i} + \vec{p_*})\| \times \|\vec{d_j}\|} \tag{8}$$

3.4.3 Forward Propagation in CogInfer

Note that the neuron matrix P is query-specific as the applicability of each rule is subject to the input query x. As for the query-independent matrix U, only a small part of this huge matrix is activated in the forward propagation because many relational facts are irrelevant to the query as the InfergGraph indicates. Specifically, if a symptom s_j is included in the input x, all triples that led by s_j will be activated. The activation matrix I is defined as follows.

$$I = [I_1, ..., I_j, ..., I_k], I_j = \begin{cases} \vec{1} \in \mathbb{R}^n, & \text{if } s_j \in x \\ \vec{0} \in \mathbb{R}^n, & \text{otherwise} \end{cases} \tag{9}$$

By applying Hadamard product ($\odot$) on each element of U and I, we can then obtain the activated matrix U'.

$$U' = [U'_{p_1}, ..., U'_{p_i}, ..., U'_{p_t}], U'_{p_i} = U_{p_i} \odot I \tag{10}$$

The final input neuron matrix X is produced via concatenation ($\oplus$) of the rule-generated neuron matrix and activated KG-generated neuron matrix:

$$X = U' \oplus P\,. \tag{11}$$

As presented in Fig. 3, the fully connected layer directly connects every input neurons with all output neurons, i.e, the reasoning targets. In the supervised inference, each input neuron represents a specific expert rule or relational fact, hence the CogInfer can be regarded as a white-box model and we can easily find the most contributing neurons in the mode structure by analyzing the weight matrix W of the fully connected layer.

4 Empirical Study

4.1 Dataset and Evaluation Metrics

In this work, we situate the cognitive inference problem in the clinical diagnosis domain for initial study. Accordingly, the observations are symptoms of a patient and the conclusion refers to the most probable diagnosed disease. We introduce two clinical diagnosis datasets as initial test-beds for our task, namely, Muzhi and MDD, both of which are adapted from existing benchmarks for automatic diagnosis QA tasks. The statistics is presented in Table 3 and construction of dataset is detailed in Appendix B.

Models	Capability				Muzhi			MDD		
	Facts	Rules	Train.	Exp.	Hits@1	Hits@2	MRR	Hits@1	Hits@2	MRR
MAJORITYGUESS	✗	✗	✗	✓	0.225	0.465	0.496	0.065	0.065	0.225
MYCIN	✗	✓	✗	✓	0.197	0.197	0.398	0.278	0.287	0.342
PURELINK	✓	✗	✗	✓	0.563	**0.732**	0.718	0.528	0.602	0.631
COGINFER	✓	✓	✗	✓	**0.592**	0.704	**0.722**	**0.606**	**0.713**	**0.710**

Table 2: Comparison with knowledge-driven methods in unsupervised setting.

Table 3: Dataset Statistics.

Statistic	Muzhi	MDD
Samples	710	2,151
Symptoms	66	93
Diseases	4	12
Rules	92	182
Entities	19,737	293,879
Predicates	7	162
Avg. Entities / Query (Train)	5.7	5.1
Avg. Entities / Query (Test)	4.9	5.3

Depending on the availability of training examples, we adopt different evaluation metrics for unsupervised and supervised settings, respectively. Specifically, we use Hits@k (k=1,2) and mean reciprocal rank (MRR) as the main evaluation metrics. Additionally, we also plot the Accuracy-Coverage curve to evaluate the knowledge-driven models and report the macro precision (Pre.), recall (Rec.) and F1-score (F1) to evaluate the data-driven models. The design of evaluation metrics is detailed in Appendix C.

4.2 Baseline Methods

4.2.1 Knowledge-driven Methods

MAJORITYGUESS is a simple baseline for reference. MYCIN is a representative of expert systems that relies on if-then rules to perform reasoning. PURELINK is a link prediction based reasoning method, which utilizes the distributed representations of relational facts to calculate the CFs for each reasoning targets. The implementation details are described in Appendix D.

4.2.2 Data-driven Methods

We compare our method with a wide range of data-driven methods in supervised setting, including two representative statistical machine learning methods k-nearest neighbor (KNN), logistic regression (LR), one feature-selective logistic regression with lasso regularization (LASSOLR), one neural-based method Multi-layer Perceptron (MLP), and one ensemble method named explainable boosting method (EBM) (Lou et al., 2013).

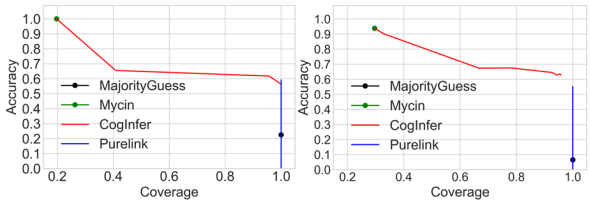


Figure 4: Accuracy-Coverage curve of knowledge-driven methods on Muzhi (left) and MDD (right).

4.3 Experimental Results

4.3.1 Comparison with Knowledge-driven Methods

Table 2 shows the performance of different knowledge-driven methods for cognitive inference problem in unsupervised setting. Specifically, for unsupervised setting, the ground-truth conclusion of each query is accessible only at the test phase for evaluation. In other words, this setting requires the reasoners not to learn from labeled examples but to make decisions merely based on knowledge, which clearly rules out the data-driven methods. Hence, it is not surprising that all the methods fail in trainability (Train.). Though there is no difference in explainablity (Exp.) among these knowledge-driven models, our method is the only one that simultaneously utilizes both expert rules and relational facts for machine reasoning. It is interesting to find that the KG-based PURELINK substantially surpass the rule-based MYCIN in both datasets, demonstrating the utilities of different representations of symbolic knowledge for the cognitive inference problem. We can also find that the performance gap between COGINFER and PURELINK in MDD dataset is much greater than that in the Muzhi dataset. We attribute this to the differences in complexity between the two datasets. More precisely, the diseases and predicates in MDD dataset is 3x and 23x as many as that in Muzhi, making the link prediction much harder for PURELINK. Nonetheless, the increased complexity from Muzhi to MDD even leads to a slight performance rise (0.592 → 0.606 in terms of Hits@1) for COGINFER, indicating that our method is more suitable

	Models	Capability				Muzhi						MDD					
		Facts	Rules	Train.	Exp.	Pre.	Rec.	F1	Hits@1	Hits@2	MRR	Pre.	Rec.	F1	Hits@1	Hits@2	MRR
W/o CogKG	KNN	✗	✗	✓	✗	0.651	0.637	0.615	0.592	0.915	0.776	0.808	0.805	0.798	0.787	0.870	0.851
	EBM	✗	✗	✓	✓	0.707	0.707	0.697	0.690	1.0	0.845	0.823	0.818	0.813	0.810	0.912	0.883
	MLP	✗	✗	✓	✗	0.750	0.741	0.729	0.718	0.986	0.857	0.833	0.835	0.829	0.829	0.903	0.890
	LASSOLR	✗	✗	✓	✗	0.777	0.776	0.769	0.761	0.986	0.878	0.832	0.834	0.828	0.829	0.921	0.894
	LR	✗	✗	✓	✗	0.782	0.769	0.769	0.761	0.972	0.876	0.842	0.839	0.833	0.833	0.931	0.897
W/ CogKG	COGINFER	✓	✓	✓	✓	**0.820**	**0.811**	**0.797**	**0.789**	**1.0**	**0.894**	**0.877**	**0.861**	**0.857**	**0.856**	**0.931**	**0.908**

Table 4: Comparison with data-driven methods in supervised setting.

and effective for the complex scenarios.

We also plot the accuracy-coverage curve of knowledge-driven methods in Fig. 4. A method cannot "cover" a test query if the query activates none of its rules and therefore the method cannot reach any conclusion. The brittle rule-based method MYCIN achieves high accuracy at a low coverage. It works almost perfectly on a small percentage of test queries (20% in Muzhi; 30% in MDD) where at least one of its inference rules is activated, but fails completely on the remaining test queries where none of its rules can be applied.

In contrast, as we change the threshold for link prediction in PURELINK, the accuracy-coverage data points surprisingly present a vertical line instead of a curve. In other words, despite it might have a limited performance in accuracy, it consistently reaches a perfect score of 1.0 in coverage, showing the strong generalizability of distributed representations of KG. Similarly, as we change the threshold for link prediction in COGINFER, the curve always starts from exactly where the MYCIN lies. **This implies that the rule-based MYCIN is a special case of the proposed COGINFER.** Specifically, as stated in Section 3.3, when the threshold exceed a certain value, the Line $6 \sim 7$ in Alg. 3 will be disabled and the reaming part performs the same steps as the expert system. Generally, it can be observed that the accuracy and coverage constrain each other on both datasets, but we can always find a reasonable balance between the two metrics, showing the flexibility of the proposed method.

4.3.2 Comparison with Data-driven Methods

Table 4 shows the performance of different data-driven methods for cognitive inference problem in supervised setting. Specifically, for supervised setting, we are provided with the labeled examples for both training and evaluation. The reasoners are free to learn from both knowledge and training data to make decisions. However, few existing data-driven methods can utilize the expert rules and relational facts for training. Among all baselines, the EBM is the only one that has explainability though its overall performance is not satisfying enough. In contrast, with the CogKG, our method COGINFER achieves collectively learning from both the symbolic knowledge and labeled data while keeping the explainability.

It can be observed that the performances of all baselines are relatively stable on the two datasets. Specifically, the KNN always give the worst performance while the LR keeps the leading position. Noticeably, the LASSOLR is a feature-selective method and is expected to be more effective than the vanilla LR. However, the performances of LASSOLR and LR are quite close to each other, implying that the symptoms in the original feature space leave much to be desired in separability. As presented in Section 3.4, we argue that such knowledge-generated features make it much easier for the optimizer to reach the global optimum as the knowledge greatly enriched the original feature space and make it more separable and tend to be consistent. With sufficient training examples, the COGINFER consistently surpasses all baselines on the two datasets under both classification and ranking metrics, showing its superiority over the time-tested data-driven methods.

4.4 Ablation Study

To analyze the utility of expert rules and relational facts in the trainable implementation of COGINFER, we conduct a set of ablation study and report the F1-score, as presented in Table 5. Specifically, we train the COGINFER with only KG-generated features and Rule-generated features, respectively. Moreover, as we adopt pre-trained embeddings in the embedding layer, we also investigate its utility in the model. Generally, the model will gain additional performance boost after fine-tuning the embeddings, indicating the importance of adjusting the general embedding to task-specific embedding. Note that the fine-tuning does not affect the performance of models with only rule-generated features because such features are determined before the training process.

On the other hand, we can find that both the KG-generated features and rule-generated features

Table 5: Ablation Study. F1-score is reported and FE indicates fine-tuning embeddings.

Input	Muzhi	MDD
All Features (w/ FE)	0.797	0.857
All Features (w/o FE)	0.730	0.850
KG Feature (w/ FE)	0.711	0.759
KG Feature (w/o FE)	0.627	0.717
Rule Feature (w/ FE)	0.362	0.366
Rule Feature (w/o FE)	0.362	0.366

contribute a lot to the effectiveness of the proposed method. As presented above, the KG Feature is comparatively more influential than the Rule Feature. We attribute this to the differences in feature size as the KG-generated features are generally multiple times of the rule-generated features. Moreover, as each feature corresponds to an explicit semantic meaning, we also conduct interpretability analysis (see Appendix E) to further investigate the learned model.

5 Conclusion

In this work, we introduce a new machine reasoning task, namely, cognitive inference problem, which directly challenges existing knowledge-driven and data-driven methods. To address this problem, we also introduce the cognitive knowledge graph (CogKG) that aims to unify the heterogeneous symbolic knowledge of expert rules and relational facts in knowledge graph, and propose a general framework COGINFER with two implementations. Experimental results on two clinical diagnosis benchmarks demonstrate the superiority of our work over existing methods.

Acknowledgements

The authors would like to thank the anonymous referees for their valuable comments. This work is supported by the National Natural Science Foundation of China (No.61976102 and No.U19A2065), the Science and Technology Development Program of Jilin Province (No.20210508060RQ) and the Fundamental Research Funds for the Central Universities, Jilin University.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *NIPS*, 26.

Bruce G Buchanan and Edward H Shortliffe. 1984. *Rule-based expert systems: the MYCIN experiments of the Stanford Heuristic Programming Project*. Addison-Wesley.

Odmaa Byambasuren, Yunfei Yang, Zhifang Sui, Damai Dai, Baobao Chang, Sujian Li, and Zan Hongying. 2019. Preliminary study on the construction of chinese medical knowledge graph. *Journal of Chinese Information Processing*, 33(10):1.

Vesna Cukic, Vladimir Lovre, Dejan Dragisic, and Aida Ustamujic. 2012. Asthma and chronic obstructive pulmonary disease (copd)–differences and similarities. *Materia socio-medica*, 24(2):100.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, Tim Finin, et al. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*.

Edward A Feigenbaum. 1980. Knowledge engineering: The applied side of artificial intelligence. Technical report, Department of Computer Science, Stanford University.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly embedding knowledge graphs and logical rules. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 192–202.

Jiawei Han, Jian Pei, and Yiwen Yin. 2000. Mining frequent patterns without candidate generation. *ACM sigmod record*, 29(2):1–12.

Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou, and Maosong Sun. 2020. More data, more relations, more context and more openness: A review and outlook for relation extraction. In *Proceedings of the 10th International Joint Conference on Natural Language Processing*, pages 745–758.

Ian Horrocks, Peter F Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, Mike Dean, et al. 2004. Swrl: A semantic web rule language combining owl and ruleml. *W3C Member submission*, 21(79):1–31.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*.

Hassan Khosravi, Oliver Schulte, Tong Man, Xiaoyuan Xu, and Bahareh Bina. 2010. Structure learning for markov logic networks with many descriptive attributes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pages 487–493.

Douglas B Lenat. 1995. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

David Liben-Nowell and Jon Kleinberg. 2003. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. 2013. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–631.

Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. 2019. Anytime bottom-up rule learning for knowledge graph completion. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3137–3143.

Pasquale Minervini, Matko Bošnjak, Tim Rocktäschel, Sebastian Riedel, and Edward Grefenstette. 2020. Differentiable reasoning on large knowledge bases and natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5182–5190.

Marvin Minsky and Seymour Papert. 1969. An introduction to computational geometry. *Cambridge tiass., HIT*.

Stephen Muggleton and Luc De Raedt. 1994. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19:629–679.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

JR Quinlan. 1987. Generating production rules from decision trees. In *Proceedings of the 10th international joint conference on Artificial intelligence-Volume 1*, pages 304–307.

Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine learning*, 62(1):107–136.

Tim Rocktäschel and Sebastian Riedel. 2017. End-to-end differentiable proving. *Advances in neural information processing systems*, 30.

Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. 2015. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of the 2015 conference of the north American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1119–1129.

Amit Singhal. 2012. Introducing the knowledge graph: things, not strings. Google Official Blog.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv:1904.09223*.

Geoffrey G Towell and Jude W Shavlik. 1994. Knowledge-based artificial neural networks. *Artificial intelligence*, 70(1-2):119–165.

Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Michal Walczak, Julius Pfrommer, Annika Pick, et al. 2021. Informed machine learning-a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*.

Zhongyu Wei, Qianlong Liu, Baolin Peng, Huaixiao Tou, Ting Chen, Xuanjing Huang, Kam-fai Wong, and Xiangying Dai. 2018. Task-oriented dialogue system for automatic diagnosis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Tingyu Xia, Yue Wang, Yuan Tian, and Yi Chang. 2021. Using prior knowledge to guide bert's attention in semantic textual matching tasks. In *Proceedings of the Web Conference 2021*, pages 2466–2475.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158.

Zhiwei Yang, Hechang Chen, Jiawei Zhang, Jing Ma, and Yi Chang. 2020. Attention-based multi-level feature fusion for named entity recognition. In *International Joint Conference on Artificial Intelligence*.

Erxin Yu, Wenjuan Han, Yuan Tian, and Yi Chang. 2020. ToHRE: A top-down classification strategy with hierarchical bag representation for distantly supervised relation extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1665–1676, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhiyuan Zhang, Xiaoqian Liu, Yi Zhang, Qi Su, Xu Sun, and Bin He. 2020. Pretrain-KGE: Learning knowledge representation from pretrained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 259–266, Online. Association for Computational Linguistics.

A Detailed Discussion of Differences with Existing Works

A.1 Knowledge Representation and Reasoning

As surveyed in the recent study on integrating prior knowledge into learning systems (von Rueden et al., 2021), various sources of prior knowledge have been investigated in different representation forms, including algebraic equations (scientific knowledge), spatial invariances (world knowledge), human feedback (expert knowledge), etc. In our cognitive inference problem, we mainly focus on the symbolic representation of knowledge, i.e, expert rule and knowledge graph. Early works on artificial intelligence compose the human knowledge as discrete symbols, and introduces the traditional symbolic knowledge representation, i.e., rules, to perform complex inference (Minsky and Papert, 1969; Lenat, 1995). Generally, these time-tested rule-based methods such as expert system have achieved great success in specialized domains, but are also limited to human effort and fail to generalize due to expensive costs (Buchanan and Shortliffe, 1984). In contrast, the recently introduced knowledge graph (KG) (Singhal, 2012), as a novel symbolic knowledge representation, is regarded quite promising to overcome such bottleneck. Specifically, the rapid development of computational hardware and deep learning makes it possible to model the rich semantic connections between massive discrete knowledge represented in KG, and the symbolic knowledge of relational facts can be mapped to distributed representation, i.e., continuous embeddings (Bordes et al., 2013). Though recent pre-trained language models such as BERT (Devlin et al., 2019), ERNIE (Sun et al., 2019) show promising performance by harvesting prior knowledge in distributed representation from large-scale corpus and knowledge graph, they all require a large amount of data and computational resources for fine-tuning and thus can only partially address the cognitive inference problem.

Despite there are a few attempts to combine first-order logic and relational facts for machine reasoning, they only focus on fixed compositional patterns of predicates (Horrocks et al., 2004; Rocktäschel et al., 2015; Guo et al., 2016; Rocktäschel and Riedel, 2017; Meilicke et al., 2019; Minervini et al., 2020). Therefore, they are strictly limited to tasks that merely reason about multi-relational data such as knowledge graph completion and relation classification, instead of inference problems where observations lead to conclusions. To the best of our knowledge, we are the first to integrate the expert rules into knowledge graph with a unified knowledge representation framework for such complex machine reasoning task.

A.2 Integrating Knowledge into Learning Systems

Different from the typical knowledge-driven artificial intelligence (AI) such as expert system, the data-driven AI such as machine learning (ML) is believed to be more generalizable due to its capability of learning implicit knowledge from labeled data, alleviating the knowledge acquisition bottleneck (Feigenbaum, 1980). However, the ML systems are substantially subject to the availability of training data and are quite limited in some cases where labeled data is hard to obtain. One potential solution is to integrate prior knowledge into learning system, which is also noted as informed machine learning (von Rueden et al., 2021). To achieve this goal in our work, there are three key challenges. First, the proposed model is expected to learn from the knowledge (i.e., expert rules and relational facts) if labeled data is unavailable. Second, if trainable, the model is required to not only reason with knowledge, but also train with knowledge. Third, a unified application of knowledge in both training and inference is anticipated.

Previous works intergating prior knowledge includes: (1) adding knowledge into learning objective (e.g., knowledge as regularization), but knowledge itself is not a part of the model. For instance, Xia et al. use prior knowledge to guide the attention matrix in BERT (Xia et al., 2021); (2) using knowledge as parameter initialization. For example, Zhang et al. proposed to first learn entity and relation representations via pre-trained language models and then use this prior knowledge (i.e., the learned representations) to initialize the knowledge graph embedding models (Zhang et al., 2020); (3) using knowledge as model architecture. Typical models include inductive logic programming (ILP) (Muggleton and De Raedt, 1994), Markov logic network (MLN) (Richardson and Domingos, 2006), and knowledge-based artificial neural networks (KBANN) (Towell and Shavlik, 1994), etc. However, these methods only focus on the logic rule (inference principle), neglecting the rich relational facts in knowledge graph.

Though these attempts achieved preliminary success, none of them can directly integrate both expert rules and relational facts in existing KG into the learning system, and they can only partially address the above challenges, which greatly motivates our work.

B Dataset Construction

To prepare the preliminaries of our task, we first harvest labeled examples from two dialogue datasets (namely, Muzhi and MDD) that are originally used for automatic diagnosis, in which the symptoms as features and the diagnosed disease as label. The relational facts are directly collected from existing well-constructed knowledge graphs like Chinese Medical Knowledge Graph (CMeKG) (Byambasuren et al., 2019) and SNOMED-CT[6], accompanying with Muzhi (Chinese) and MDD (English) respectively. We further apply association rule mining on the labeled data followed by human expert validation to craft if-then rules. Specifically, we invite three medical experts to check the mined rules and filter them via consistency validation. Lastly, we also manually align the entities in premise and conclusion of each rule to the terminologies of KG to make the expert rules and relational facts compatible with each other. The details of each dataset are as follows.

- **Muzhi** dataset is originated from a Chinese online healthcare community[7] and is firstly used for dialogical automated diagnosis (Wei et al., 2018). In this work, we collect the explicit symptoms and implicit symptoms as observations and the diagnosed disease as conclusion to create labelled examples. After terminology alignment, it contains 710 samples with 66 symptoms related to 4 diseases, i.e., infantile diarrhea (ID), children functional dyspepsia (CFD), upper respiratory infection (URI), and children's bronchitis (CB). We randomly split the dataset to training set, validation set, test set in the proportion of 8:1:1. Additionally, this dataset contains 92 if-then rules related to the above 4 diseases, and 19,737 distinct entities connected with 7 predicates.

- **MDD** is an English medical diagnosis dialogue (MDD) dataset proposed in the ICLR 2021 challenge[8]. Following the Muzhi dataset, the original dialogical records are converted into labeled instances. After terminology alignment, the MDD dataset is three times larger than the Muzhi dataset, containing 2,151 samples with 93 symptoms related to 12 diseases. Likewise, the dataset is randomly split to 8:1:1 for training, validation and test. It contains 182 if-then rules related to the above 12 diseases, and 293,879 distinct entities connected with 162 predicates.

C Design of Evaluation Metrics

As the proposed COGINFER is not bound with unsupervised inference or supervised inference, we can evaluate it under both unsupervised setting and supervised setting. According to the truthiness of the most likely conclusion of each query, the result can be categorized into three types, namely, true conclusion (TC), false conclusion (FC), and not conclusive (NC), indicating the model cannot output any conclusion for the query. Generally, we evaluate the model with the following two ranking metrics, i.e., Hits@k (k=1,2) and mean reciprocal rank (MRR).

$$Hits@k = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \mathbb{I}[r \leq k] \tag{12}$$

$$MRR = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} r^{-1} = \left(\frac{|\mathcal{R}|}{\sum_{r \in \mathcal{R}} r^{-1}}\right)^{-1} \tag{13}$$

where $\mathcal{R}$ denotes the set of ranks for all predicted most likely conclusions and $\mathbb{I}$ is the indicator function. More precisely, for each TC result, the rank will always be 1. For any FC result, the rank could be any integer in $[2, n]$, where n is the number of diseases contained in the dataset. For NC result, the rank is set to the worst case by default, i.e, it will always be n.

In particular, for knowledge-driven method, when it encounters a query for which no rules or facts have been pre-defined in the knowledge base, the system will get stuck and cannot output any conclusion. In other words, it does not even understand the query. Therefore, we also define *Accuracy* and *Coverage* to evaluate these knowledge-driven mod-

[6] https://www.nlm.nih.gov/healthit/snomedct

[7] http://muzhi.baidu.com

[8] https://mlpcp21.github.io/pages/challenge.html

els.

$$Accuracy = \frac{TC}{TC + FC} \tag{14}$$

$$Coverage = \frac{TC + FC}{TC + FC + NC} \tag{15}$$

Instead of reporting the discrete data points, we plot the Accuracy-Coverage curve to comprehensively compare different knowledge-driven models.

In contrast, for data-driven models, they have predefined fixed reasoning targets and will not output any NC results. Hence, the above *Accuracy* and *Coverage* metrics cannot properly evaluate the data-driven models. In this case, we adopt the classification metrics such as macro precision (Pre.), recall (Rec.) and F1-score (F1) as the evaluation metrics.

D Implementation Details

For knowledge-driven baselines, the MYCIN is implemented with the *Paip-python* library[9]. The threshold for link prediction in PURELINK is determined from [-1, 1] via grid search on the mixed set of training and validation data. For data-driven baselines, we implement the EBM with the *InterpretML* library[10]. The rest data-driven baselines are implemented with the *scikit-learn* package (Pedregosa et al., 2011). All hyperparameters are kept as default except the following ones that are determined through grid search on validation set. For KNN, the number of neighbors k is searched from [1, 2, 3, ..., 10]. For LR and LASSOLR, the inverse of regularization strength C is searched from [0.1, 1, 10]. For MLP and EBM, the learning rate is searched from [1e-5, 1e-4, 1e-3, 1e-2]. Moreover, the number of hidden layers/nodes in MLP is searched from [(100,), (50,50), (50,50,50)].

For the proposed COGINFER, the threshold for link prediction in non-trainable (unsupervised) implementation is determined from [-1, 1] via grid search on the mixed set of training and validation data. Note that the trainable implementation of our framework can be initialized from pre-trained embeddings and weights. For Muzhi dataset, the embedding layer is initialized with pre-learned embeddings of CMeKG by TransE and the fully connection layer is initialized with that of a pre-trained group-lasso regularised logistic regression model. For MDD dataset, the embedding layer is initialized with pre-learned embeddings of SNOMED-CT by TransE and the fully connection layer is initialized with that of a pre-trained L2 regularised logistic regression model. For both datasets, we only fine-tune the embedding layer while freezing the fully connected layer when performing optimization.

E Interpretability Analysis

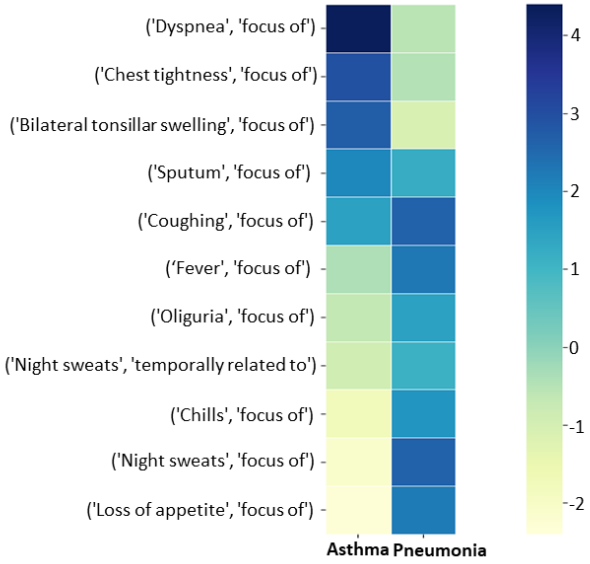
To illustrate the interpretability of the proposed COGINFER, we conduct two sets of case study as presented in Fig. 5. The values in global weight (Fig. 5(a)) are selected from columns of the weight matrix W, corresponding to two closely-related respiratory diseases “Asthma" and “Pneumonia". Likewise, the values in instance-level behavior (Fig. 5(b)) are selected from columns of the production of input neuron matrix X and weight matrix W.

For the global weight, we visualize some selected cells of two rows (corresponding to the disease) of the weight matrix W to interpret the reasoning of COGINFER. According to the heatmap, the most effective rules and facts for diagnosing “Asthma” includes “Dyspnea”, “Chest tightness” and so on. In contrast, the diagnosis of “pneumonia” mainly involves “Night sweats”, “Loss of appetite” and “Chills”, which helps distinguish “Asthma” from “Pneumonia”. Meanwhile, we can learn that “Sputum” and “Coughing” are similar symptoms of “Asthma” and “Pneumonia”. Encouragingly, according to the public literature, what we learn from the weight matrix is consistent with common medical knowledge (Cukic et al., 2012).

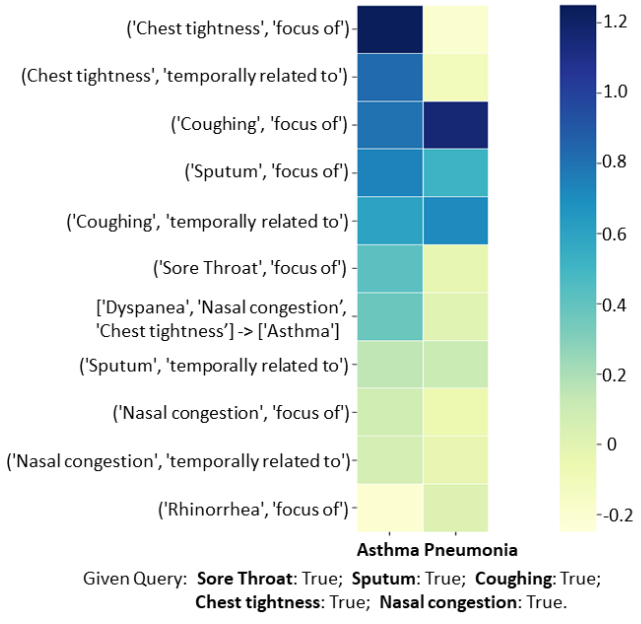
For the instance-level behavior, we study a specific sample given its symptoms. We visualize some cells of the production of input neuron matrix X and weight matrix S. These scores represent the importance of the corresponding activated rules or facts in the diagnostic process. As “coughing” and “Sputum” are common symptoms of “Asthma” and “Pneumonia”, they both score high under the corresponding rules. Moreover, it is interesting to find that the high scores of “Chest tightness” and “Sore throat” are also in line with the fact that they are widely believed to be “Asthma”-indicative symptoms that lead to the diagnosis of “Asthma”, revealing the interpretability of the proposed method.

[9] https://github.com/dhconnelly/paip-python

[10] https://github.com/interpretml/interpret



(a) Global Weights



(b) Instance-level Behavior

Figure 5: Interpretability study with real cases.