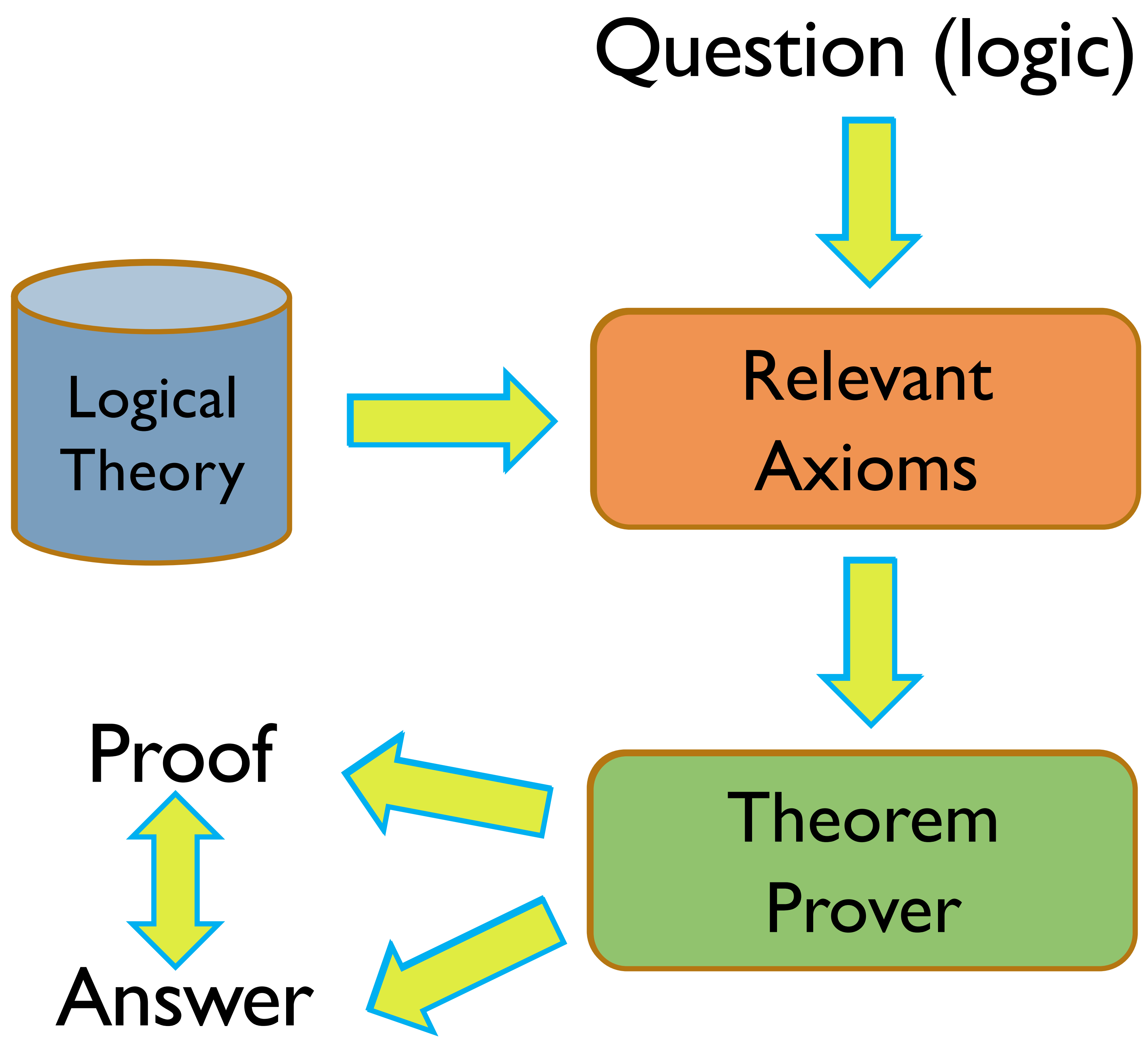
Probabilistic Graph Reasoning for Natural Proof Generation

Changzhi Sun Xinbo Zhang Jiangjie Chen Chun Gan
Yuanbin Wu Jiaze Chen Hao Zhou Lei Li

ByteDance 字节跳动
复旦大學 FUDAN UNIVERSITY
華東師範大學 EAST CHINA NORMAL UNIVERSITY
WISCONSIN UNIVERSITY OF WISCONSIN–MADISON

Reasoning Over Formal Representation

- Pros
  - Interpretable
  - Easy combine human knowledge
- Cons
  - Knowledge acquisition bottleneck
  - Brittleness
    - when confront with unusual or atypical cases



Formal Reasoning

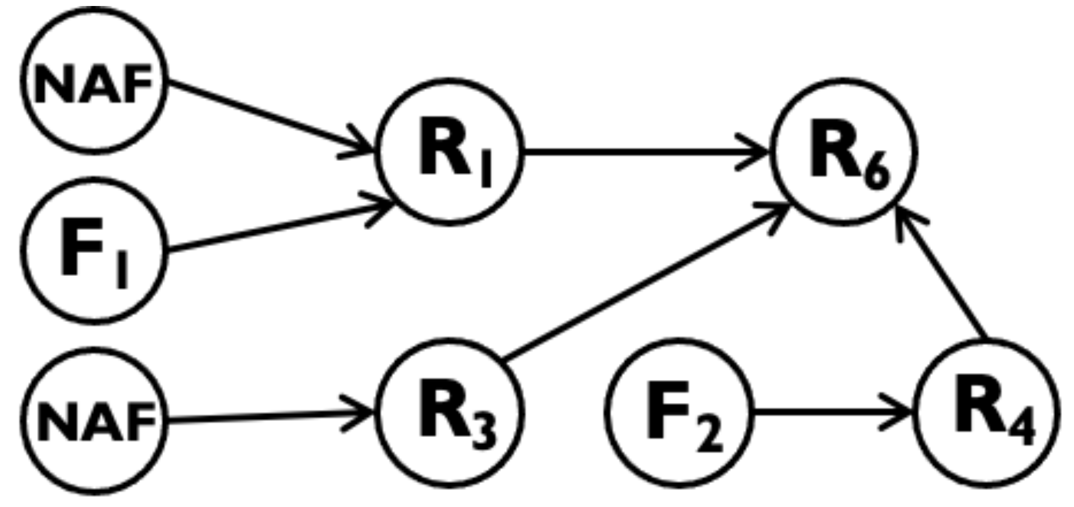
[Saha+ 2020, Musen&Lei 1988]

Reasoning over Natural Language

- **Input**: a set of facts and rules and a question expressed in natural language.
- **Output**:predict the answer and provide proof to prove or disprove the question.
- **Proof:**
  - Node: fact, rule or NAF
  - Edge: logical deduction
- **Potential advantages**
  - Write theories in natural language
  - Have the machine apply general knowledge

closed-world assumption

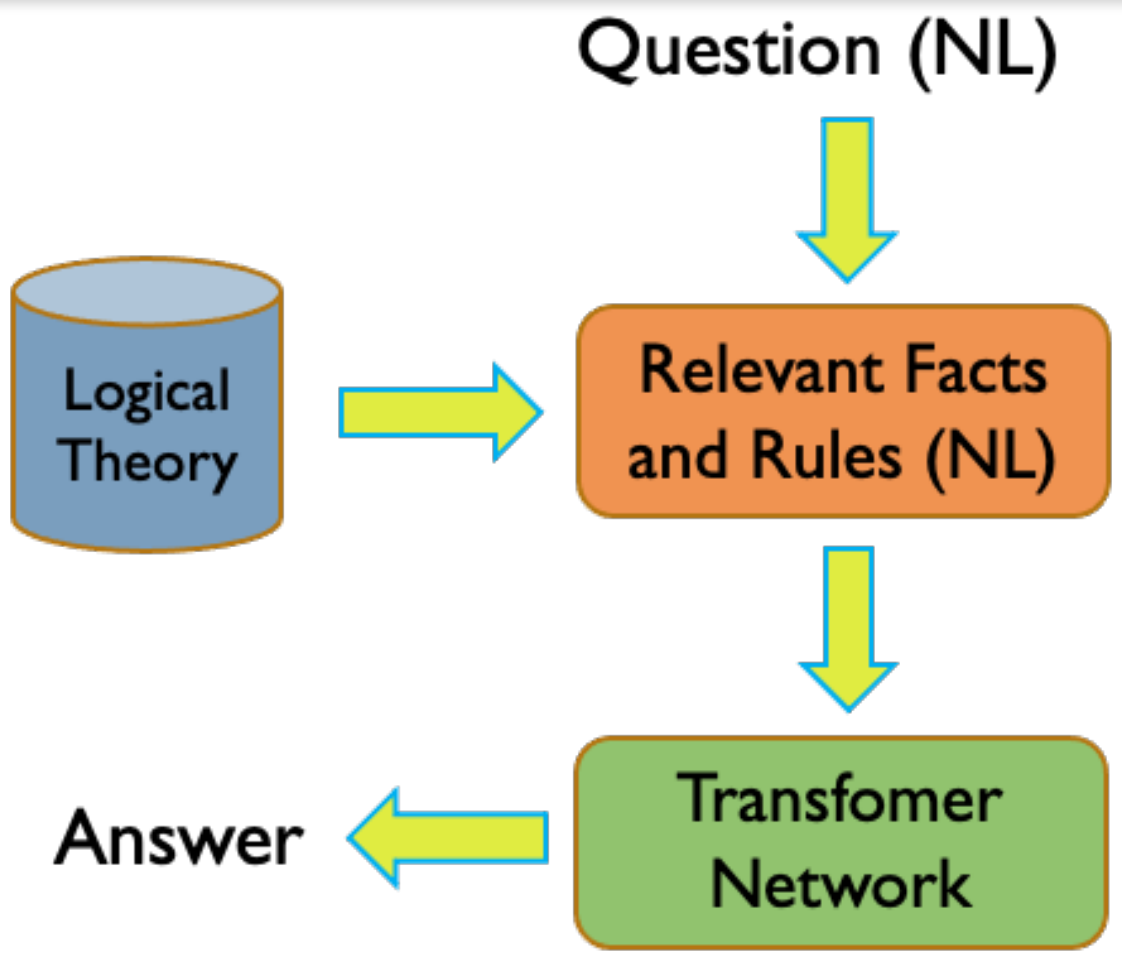
Facts :
F_1: The circuit includes the battery.
F_2: The wire is metal.
F_3: The circuit includes the bell.
Rules :
R_1: If the circuit includes the battery and the battery is not flat then the circuit is powered.
R_2: If the circuit includes the switch and the switch is on then the circuit is complete.
R_3: If the circuit does not have the switch then the circuit is complete.
R_4: If the wire is metal then the wire is conducting.
R_5: If the wire is plastic then the wire is not conducting.
R_6: If the circuit is powered and the circuit is complete and the wire is conducting then the current runs through the circuit.
Question : The current runs through the circuit.
Answer : True
Proof :



[Clark+ 2020, Saha+ 2020]

Existing Solution

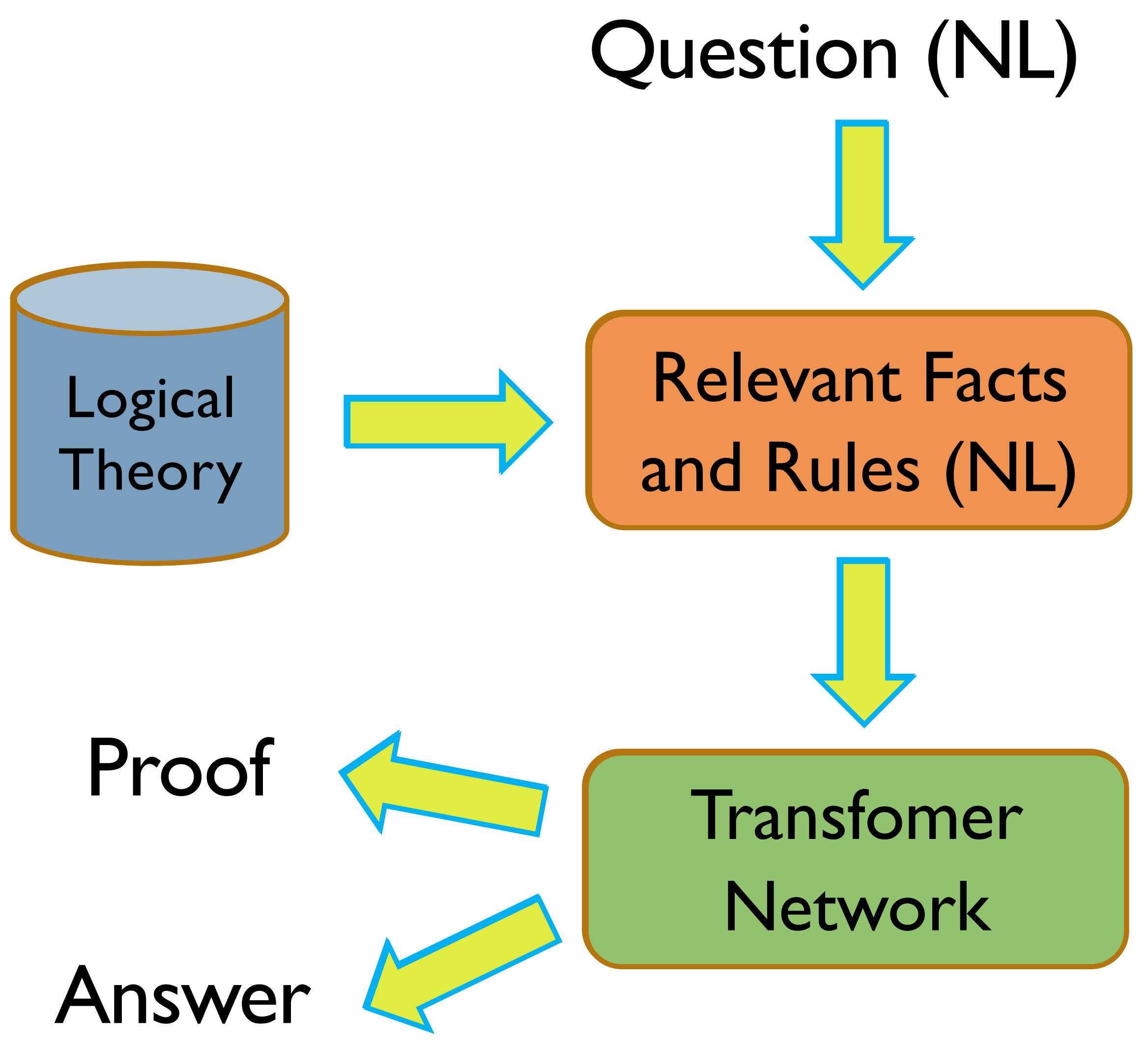
- **Soft Reasoner** [Clark+ 2020]
  - Answer prediction (0/1)
  - No proof



- **PRover** [Saha+ 2020]
  - Three sub-task, multi-task learning
    - Answer prediction (0/1)
    - Node prediction (0/1)
    - Edge prediction (0/1)

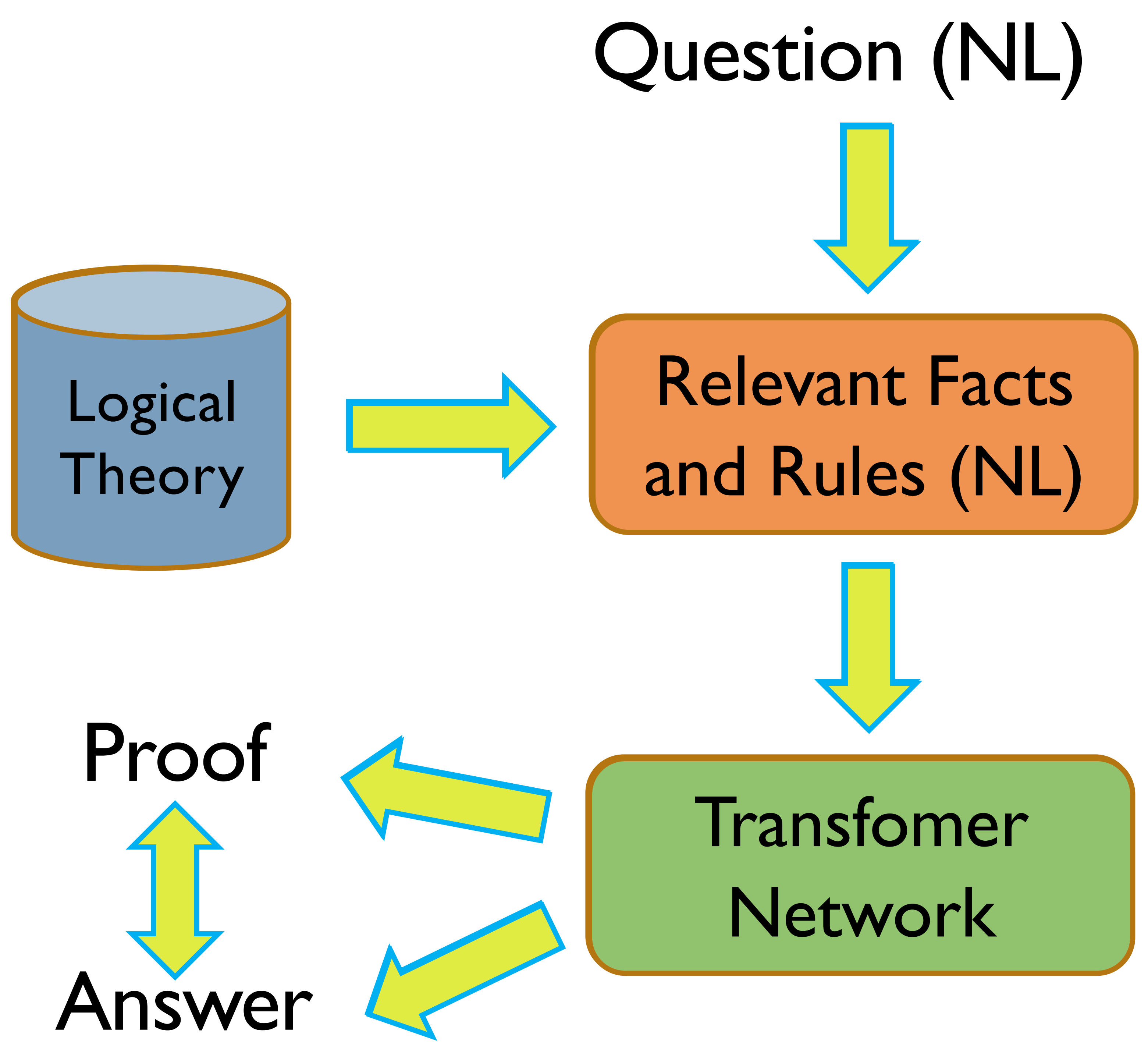
indicates whether the node/edge appears in the gold proof

  - Nodes, edges and answer are **independent** on each other



Our Solution

- **PRobr**
  - Probabilistic graphical model
  - Nodes, edges and answer are **dependent** on each other
  - Learning by variational approximation



Probabilistic Formulation

all variables

answer (0/1)

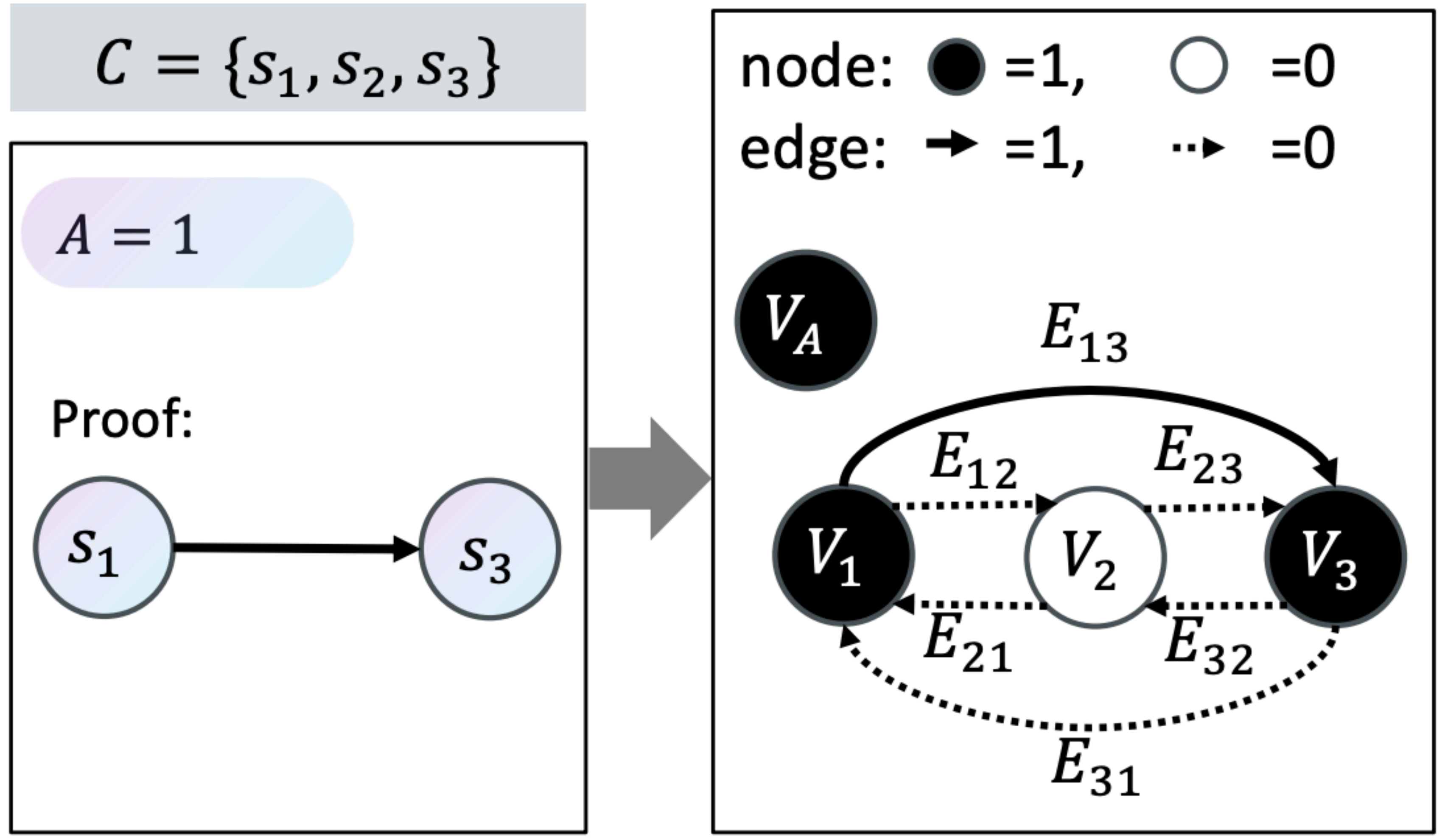
node(0/1)

edge(0/1)

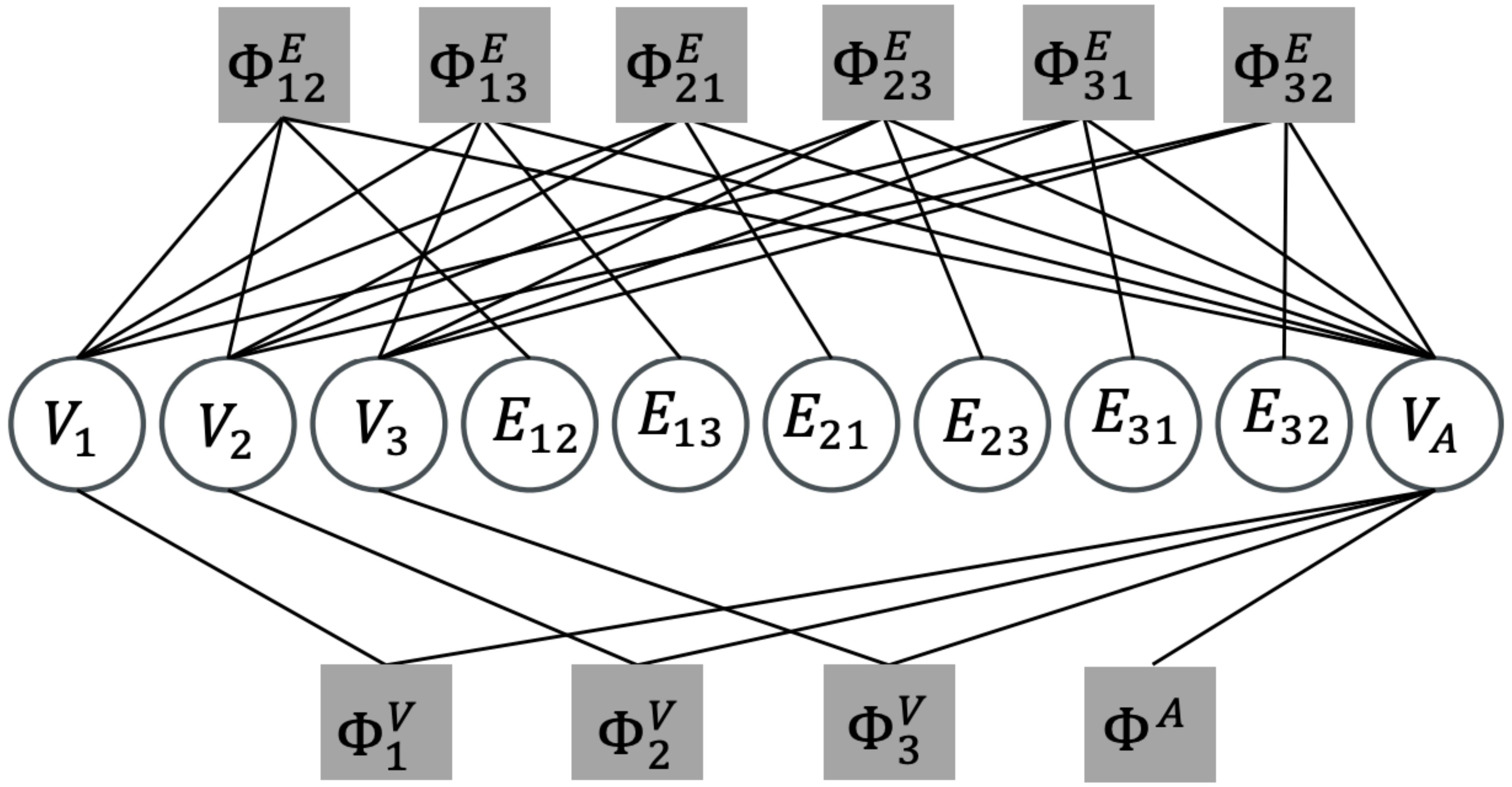
$$P(Y=y) \propto \Phi^A(a) \prod_i \boxed{\Phi_i^V(v_i, a)} \prod_{i,j} \boxed{\Phi_{ij}^E(v_i, v_j, e_{ij}, a)}$$

dependency between answer and proof node

dependency between answer, proof node and proof edge

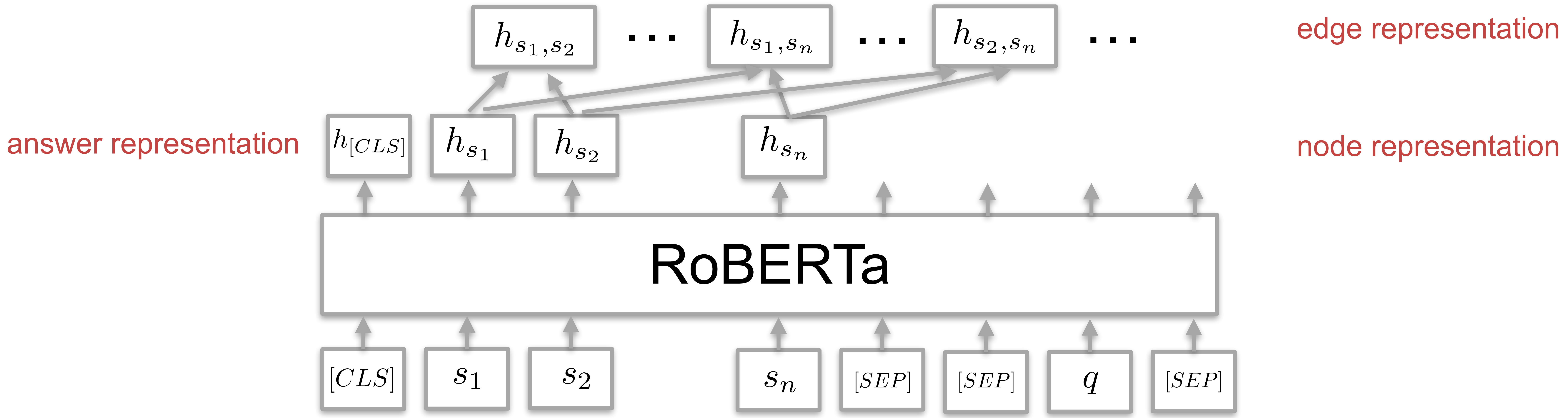


(a) Proof graph and its induced random variables.



(b) Factor graph induced by the proof graph.

Text Representation Network



h_{s_i} mean pooling over all token of sentence s_i

$$h_{s_i,s_j} = h_{s_i} \oplus h_{s_j} \oplus (h_{s_i} - h_{s_j})$$

Parameterization

Potential Function for Φ^A

$$\begin{bmatrix} \Phi^A(A=0) \\ \Phi^A(A=1) \end{bmatrix} = \mathrm{MLP}_1(h_{\texttt{[CLS]}}) \in \mathbb{R}^2$$

Potential Function for Φ^V

$$\begin{bmatrix} \Phi_i^V(V_i=0, A=0) \\ \Phi_i^V(V_i=0, A=1) \\ \Phi_i^V(V_i=1, A=0) \\ \Phi_i^V(V_i=1, A=1) \end{bmatrix} = \mathrm{MLP}_2(h_{s_i}) \in \mathbb{R}^4$$

Potential Function for Φ^E

$$\begin{bmatrix} \Phi_{ij}^E \begin{pmatrix} V_i=0, V_j=0, \\ E_{ij}=0, A=0 \end{pmatrix} \\ \vdots \\ \Phi_{ij}^E \begin{pmatrix} V_i=1, V_j=1, \\ E_{ij}=1, A=1 \end{pmatrix} \end{bmatrix} = \mathrm{MLP}_3(h_{s_i,s_j}) \in \mathbb{R}^{16}$$

Learning

- First approximation
  - Joint-likelihood $\rightarrow$ Pseudo-likelihood

$$\mathcal{L}_{\text{joint}} = -\log p(Y = y^*) \rightarrow p_{\text{pseduo}}(Y) = \prod_{y \in Y} p(y|Y_{-y}) = p(A|\mathcal{E}, \mathcal{V}) \prod_{i} p(V_i|Y_{-V_i}) \prod_{i,j} p(E_{ij}|Y_{-E_{ij}})$$

normalization constant is hard to calculate due to high-order factors of large size

1. easy to calculate when neighbors are given.
2. traditional decoding methods are based on sampling (inefficient).
3. we choose a modern approach using variational approximation.

- Second approximation
  - Pseudo-likelihood $\rightarrow$ Variational approximation

Learning

- Second approximation
  - Pseudo-likelihood $\rightarrow$ Variational approximation

$$p_{\text{pseduo}}(Y) = \prod_{y \in Y} p(y|Y_{-y}) \quad \rightarrow \quad q(Y) = q(A) \prod_{i} q(V_i) \prod_{i,j} q(E_{ij})$$

parameterization of q
$$\left\{ \begin{aligned} q(A) &= \text{Softmax}(\text{MLP}_4(h_{\texttt{CLS}})) \in \mathbb{R}^2 \\ q(V_i) &= \text{Softmax}(\text{MLP}_5(h_{s_i})) \in \mathbb{R}^2, \\ q(E_{ij}) &= \text{Softmax}(\text{MLP}_6(h_{s_i,s_j})) \in \mathbb{R}^2. \end{aligned} \right.$$

Update Parameters of P and Q

Note that the conditions are obtained by prediction of the variational distribution q

$$\mathcal{L}_{\text{node}} = -\sum_{i} \log q(V_i = v_i^*)$$

$$\mathcal{L}_{\text{edge}} = -\sum_{i,j} \log q(E_{ij} = e_{ij}^*)$$

$$\mathcal{L}_{\text{qa}} = -\log p(A = a^* | \hat{\mathcal{E}}, \hat{\mathcal{V}})$$

variational distribution q

fully conditional distribution p

Inference

- For nodes and edges

$$\hat{e}_{ij} = \arg\max q(E_{ij}) \qquad \hat{v}_i = \arg\max q(V_i)$$

- For answers

$$\boxed{\hat{\mathcal{E}} = \{\hat{e}_{ij}\}, \hat{\mathcal{V}} = \{\hat{v}_i\}}$$

$$\hat{a} = \arg\max p(A|\hat{\mathcal{E}}, \hat{\mathcal{V}})$$

- Integer Linear Programming (ILP)

Fully Supervised

- Metrics
  - **QA Accuracy (QA)**
  - **Proof Accuracy (PA)**
    - exactly match
  - **Full Accuracy (FA)**
    - both answer and proof
- Settings
  - Fully supervised
  - Few-shot
  - Zero shot

Reasoning depth

Soft Reasoner

PRover

PRobr

D	Cnt	QA			PA		FA	
		RT	PV	PB	PV	PB	PV	PB
0	6299	100	100	**100**	98.4	**98.4**	98.4	**98.4**
1	4434	98.4	99.0	**99.9**	93.2	**94.3**	93.1	**94.3**
2	2915	98.4	98.8	**99.9**	84.8	**86.1**	84.8	**86.1**
3	2396	98.8	99.1	**100**	80.5	**82**	80.5	**82**
4	2134	99.2	98.8	**100**	72.5	**76.1**	72.4	**76.1**
5	2003	99.8	99.3	**100**	65.1	**72.2**	65.1	**72.2**
All	20192	99.2	99.3	**99.9**	87.1	**88.8**	87.1	**88.8**

QA: PRobr ≈ PRover ≈ Soft Reasoner

PA&FA: PRobr > PRover

Few-shot & Zero-shot

Few-shot

Train Data		QA		PA		FA	
		PV	PB	PV	PB	PV	PB
	100%	99.3	**99.9**	87.1	**88.8**	87.1	**88.8**
RC	10%	94.5	**99.9**	**63.6**	60.4	**63.3**	60.4
	5%	80.6	**99.7**	34.0	**44.2**	32.1	**44.2**
	1%	70.2	**88.2**	20.0	**21.6**	15.1	**20.3**
RQ	30k	97.8	**99.9**	72.5	**86.8**	72.4	**86.8**
	10k	87.1	**99.9**	44.0	**72.4**	42.7	**72.3**
	1k	51.3	**82.1**	**28.0**	21.1	15.0	**18.4**

RC: queries from randomly reserved contexts

RQ:randomly reserved queries

QA&PA&FA: PRobr >> PRover

Zero-shot

Test	Cnt	QA			PA		FA	
		RT	PV	PB	PV	PB	PV	PB
B1	40	97.5	95.0	**100.0**	92.5	**100.0**	92.5	**100.0**
B2	40	100	95.0	**100.0**	95.0	**100.0**	95.0	**100.0**
E1	162	96.9	100	**100.0**	95.1	**97.5**	95.1	**97.5**
E2	180	98.3	100	**100.0**	91.7	**93.3**	91.7	**93.3**
E3	624	91.8	89.7	**98.2**	72.3	**79.3**	71.8	**79.3**
E4	4224	76.7	84.8	**95.6**	**80.6**	77.7	**80.6**	77.7
All	5270	80.1	86.5	**96.3**	**80.7**	79.3	**80.5**	79.3

QA: PRobr >> PRover

PA&FA: PRobr $\approx$ PRover

Generalize to Unseen Depth

testing on DU5 after training on DU0, DU1, DU2, DU3, respectively

DUd: answers require reasoning up to depth d for queries in DUd

Train Data	QA			PA		FA	
	RT	PV	PB	PV	PB	PV	PB
DU0	53.5	**68.7**	56.9	44.4	**50.7**	**42.8**	41.3
DU1	63.5	73.7	**97.7**	63.8	**63.9**	61.9	**63.9**
DU2	83.9	89.6	**99.9**	72.6	**74.5**	72.3	**74.4**
DU3	98.9	98.6	**99.9**	79.1	**83.2**	79.1	**83.2**
DU5	99.2	99.3	**99.9**	87.1	**88.8**	87.1	**88.8**

QA: PRobr >> PRover

PA&FA: PRobr $\approx$ PRover

PRobr makes better use of proof information when answering prediction

Take Away

- Feasibility of deductive reasoning on natural language
  - Validated on synthetic data only
  - Proof helps interpret

- Dependency helps generalization
  - Few-shot & Zero-shot
  - Variational approximation for graph modelling

Paper & Code