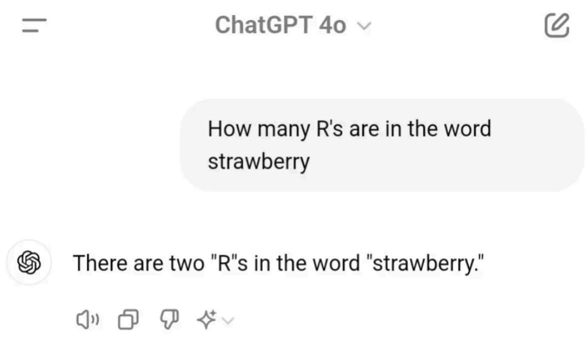
ニューロシンボリック AI における記号論理の不変性の応用

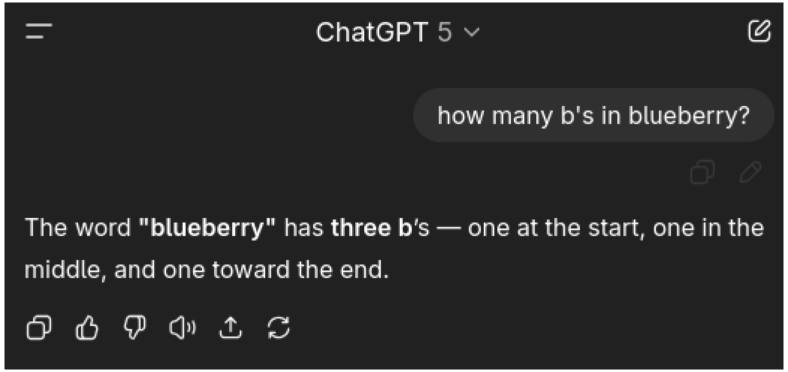
ポア　インジュン　助教

情報理工学院　村田研究室

Institute of SCIENCE TOKYO

深層機械学習の課題





- 汎化性能が不確実
- どの単語とアルファベットで間違えるかがわからない
- 単語をトークン化しているので、LLM は個々のアルファベットが見えていない
- Python などのプログラムを書けば確実に正しい回答ができる

深層機械学習とシンボリック手法

深層機械学習	シンボリック手法
◯ 実応用で優れた性能 ◯ ノイズに対して頑健	◯ 少量の学習データ ◯ 解釈可能かつ説明可能 ◯ 汎化性能の評価が可能
× 大量の学習データが必要 × 解釈可能性と説明可能性の欠如 × 汎化性能の評価ができない	× 実応用が難しい × ノイズに影響されやすい

お互いの欠点を補うことでさまざまな問題が解決できる

ニューロシンボリック AI とは？

- 深層機械学習と従来のシンボリック手法（古き良き AI）を融合する手法
- ニューロシンボリック自体は 1990 年代より研究されている[1]

DALL-E 3 より

[1]Sarker, M.K., Zhou, L., Eberhart, A. and Hitzler, P., 2022. Neuro-symbolic artificial intelligence: Current trends. Ai Communications, 34 (3), pp.197-209.

ニューロシンボリック AI の目的

深層機械学習の実データから学習する性質を利用する

↓

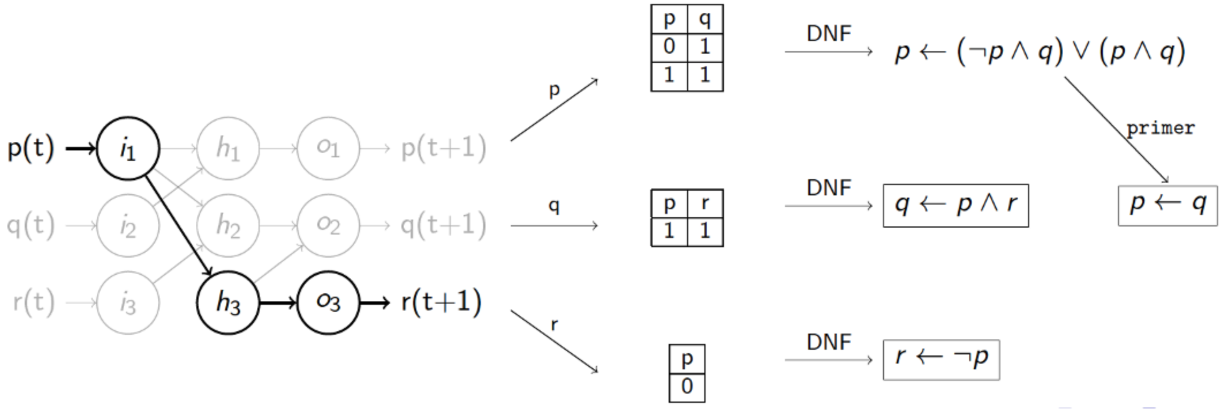
シンボリック知識を得る

ニューロシンボリックのハイブリッド手法

- Neuro[Symbolic]および Symbolic[Neuro]とも表現できる (Kautz 2020)
- 深層学習手法で推論エンジンを呼び出す (Neuro[Symbolic])
  - LINC (Olausson et al. 2023)
  - AlphaGeometry (Trinh et al. 2024)
  - ツール利用の LLM (最近リリースされている LLM のほとんどが実装済み)
- シンボリック手法から深層学習モデルを呼び出す (Symbolic[Neuro])
  - DeepProbLog (Manhaeve et al. 2018)
  - Scallop (Li et al. 2023)

ルール抽出手法

- Neuro$_{\texttt{Symbolic}}$ (Kautz 2020)
- 論理やシンボリックの性質でニューラルネットワークの構造に制約をつける
- しかし、構造に制約をかけてしまったことで深層学習の最新の技術を応用することが非常に難しい
  - C-IL^{2}P (Garcez et al. 1999)
  - NN-LFIT (Enguerrand et al. 2016)



勾配降下法

- シンボリックの問題を損失関数に変換し、勾配降下法を用いて解を求める手法
- 最終的に損失を最小化にした行列やテンソルをシンボリックな知識に変換する
- 最小化させる関数が取りうるシンボリック面での知識をすべて表現する必要があるため、大きい問題ではメモリ消費量がネックとなる
  - DFORL (Gao 2024)
  - δILP (Evans et al. 2018)

テンソル手法

- 論理演算を線形代数演算に置き換える
- 並列計算を用いてスケーラビリティ問題を解決する
  - Computation of Logic Program using Linear Algebra (Nguyen et al. 2022)
  - T-PRISM (Kojima et al. 2019)

ニューロシンボリック AI の課題のひとつ

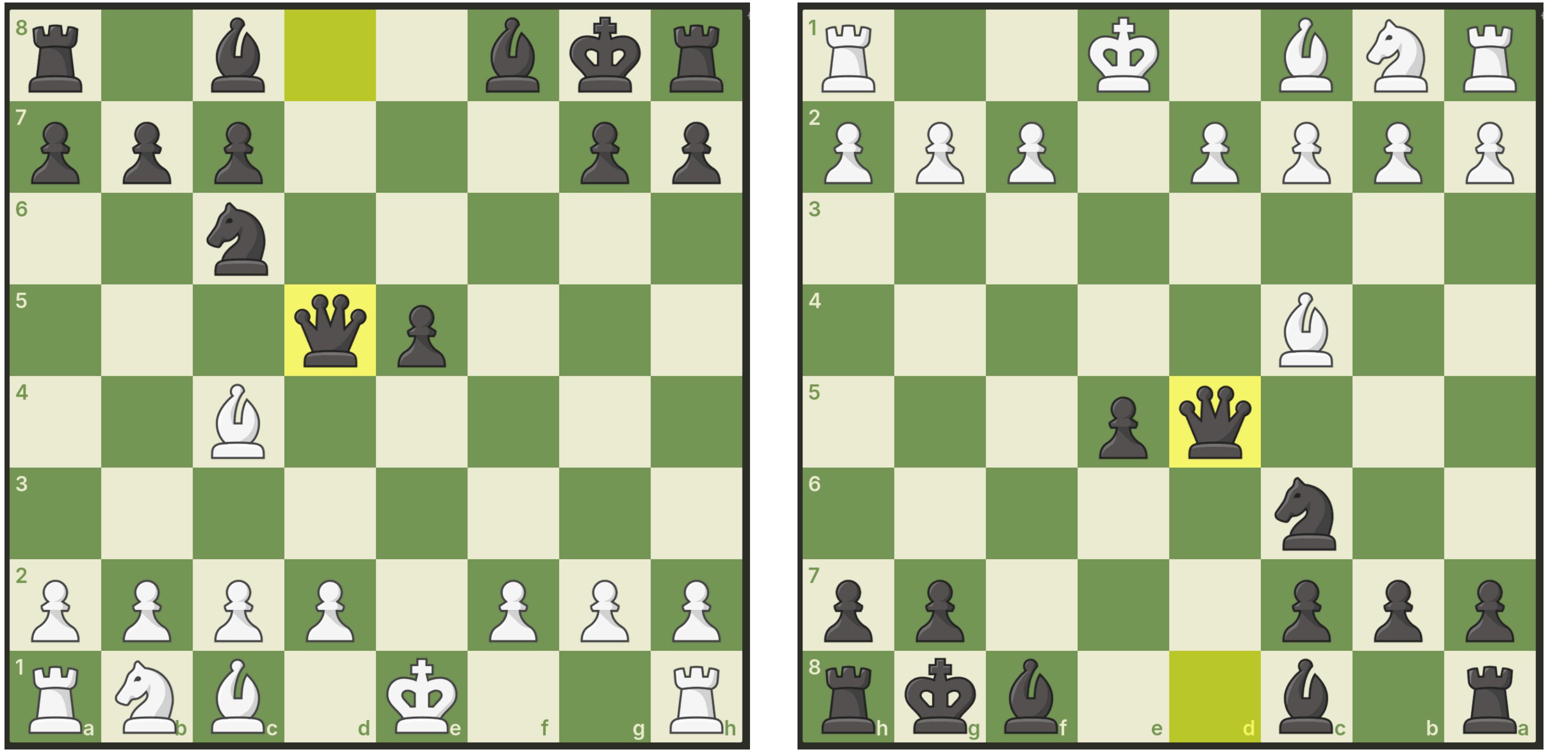
Institute of SCIENCE TOKYO

- シンボリックと連続空間(ニューラルネットワーク)の境目には組み合わせ爆発問題がある
- 連続空間では要素の順序が変わると意味合いが変わる
- シンボリックでは対称性や不変性がある
- 連続空間(ユークリッド空間)にシンボリックの対称性や不変性を組み込むにはすべての組み合わせを考慮する必要がある

不変性とは？

同じく「犬」であると分類するべき

不変性とは？



最善手が同じ

同変性とは？

8			2				5	
	6				4			
					5			
4			7			2		
	8	6						3
7		9						
		1		4		6	9	
			8		3	4		
				6				1

		1		4		6	9	
			8		3	4		
				6				1
8			2				5	
	6				4			
					5			
4			7			2		
	8	6						3
7		9						

解が似ている

不変性と同変性

インデックス$\{1, 2, ..., n\}$のすべての順列を含んだ集合をS_nとする。

不変性 (Invariance)

定義　関数$f : X^n \to Y$が$\forall \pi \in S_n, f(\pi x) = f(x)$を満たす場合のみ f は不変である。

同変性 (Equivariance)

定義　関数$g : X^n \to Y^n$が$\forall \pi \in S_n, g(\pi x) = \pi g(x)$を満たす場合のみ g は同変である。

記号論理における不変性

- 意味論的な不変性

  Φ が充足可能 $\Leftrightarrow$ Ψ が充足可能
- 交換律における不変性

  $\Gamma \wedge \Phi \equiv \Phi \wedge \Gamma$
- 単調性

  $\Gamma \models \Phi$ ならば $\Gamma \cup \Delta \models \Phi$
- 変数の命名

  $\alpha \equiv \beta$ ならば $\varphi[\alpha] \equiv \varphi[\beta]$

なぜシンボリックの不変性に注目するか？

- 必要な学習データの量が減る
- 汎化性能の向上
- 頑健性が向上する、背景など予測と無関係の変動に対応できる
- 重要な特徴のみを学習するモデルの構築につながる
- 多くのニューラルネットワークは不変性を考慮していない

不変性や同変のニューラルネットワーク

- CNN のプーリング操作により入力順序の不変性が保たれる
- sum, max, min, average などのプーリング操作[2,3]
- Alphafold の 3d 同変 Transformer[4]
- Set Transformer[5]
- Positional Encoding のない Transformer
- 入力順序に依存しない不変な操作を定義することで不変性を担保する

---

[2] Edwards, H. and Storkey, A., 2017, April. Towards a Neural Statistician. In 5th International Conference on Learning Representations (pp. 1-13).

[3] Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R.R. and Smola, A.J., 2017. Deep sets. Advances in neural information processing systems, 30.

[4] Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A. and Bridgland, A., 2021. Highly accurate protein structure prediction with AlphaFold. nature, 596(7873), pp.583-589.

[5] Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S. and Teh, Y.W., 2019, May. Set transformer: A framework for attention-based permutation-invariant neural networks. In International conference on machine learning (pp. 3744-3753). PMLR.

「論理ルール」というパターンを認識する

- 背景知識、正の例や負の例から論理プログラムを学習するいわゆるILP問題を対象としている
- 論理プログラムは解釈可能な知識となり、論理プログラムから導かれた答えは説明が可能
- 正の例や負の例を直接ニューラルネットワークに入力し、出力が論理プログラムとなるよう学習
- ひとつの学習済みモデルで多数のシンボリック問題を解くことが可能（いわゆる基盤モデル）
- 動的システムの状態遷移列から論理プログラムを学習するLFIT[6]問題を解くニューロシンボリックモデルを提案[7]

[6] Inoue, K., Ribeiro, T. and Sakama, C., 2014. Learning from interpretation transition. Machine Learning, 94(1), pp.51-79.

[7] Phua, Y.J. and Inoue, K., 2019, September. Learning logic programs from noisy state transition data. In International Conference on Inductive Logic Programming (pp. 72-80). Cham: Springer International Publishing.

データから標準論理プログラムを学習する

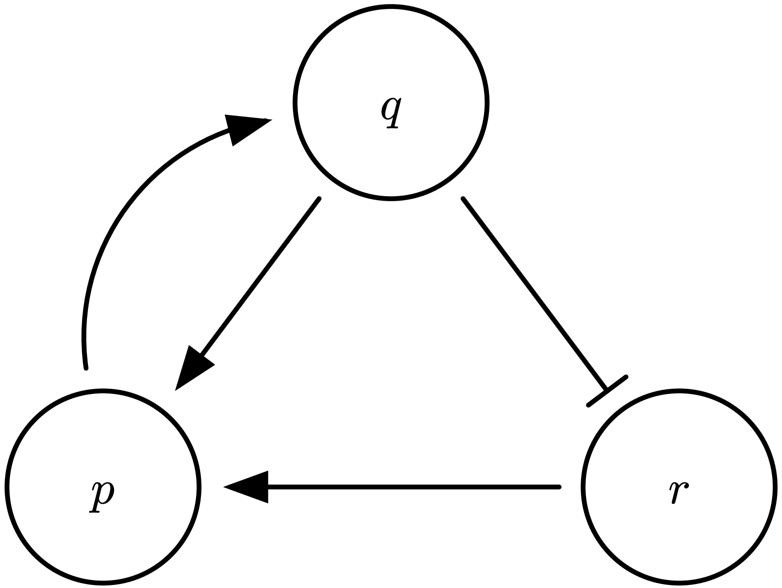
定義　論理ルール $R : A \leftarrow A_1 \wedge \ldots \wedge A_m \wedge \neg A_{m+1} \wedge \ldots \wedge \neg A_n$ の集合 P を標準論理プログラムとする。

動的システムの t 時点の状態を、集合 I で表現する。その動的システムを説明する標準論理プログラム P とする。

定義　ある時点 t での状態 I_t および標準論理プログラム P が与えられた場合、T_P 演算を用いて $t+1$ の状態 I_{t+1} が求められる。

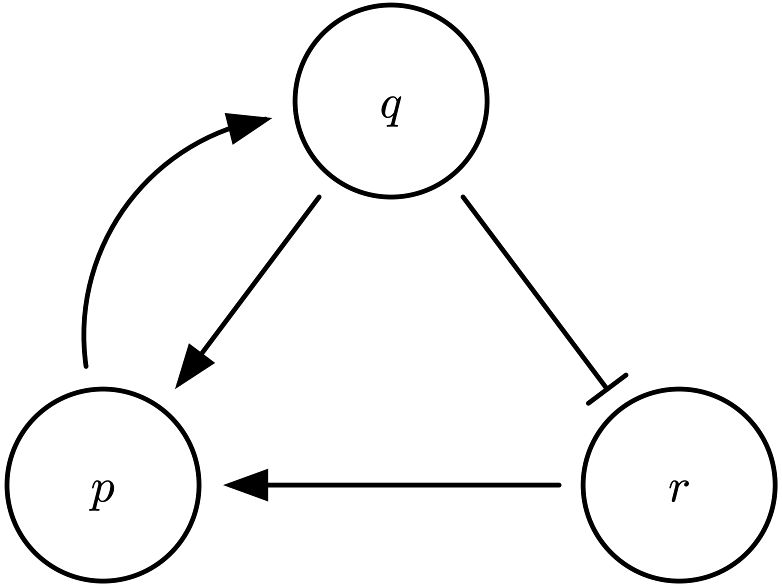
ある動的システムの状態遷移列 $I_1, I_2, \ldots, I_t$ が与えられた時、どのように P を求めるか？

例



Markov(1) の動的システム

例



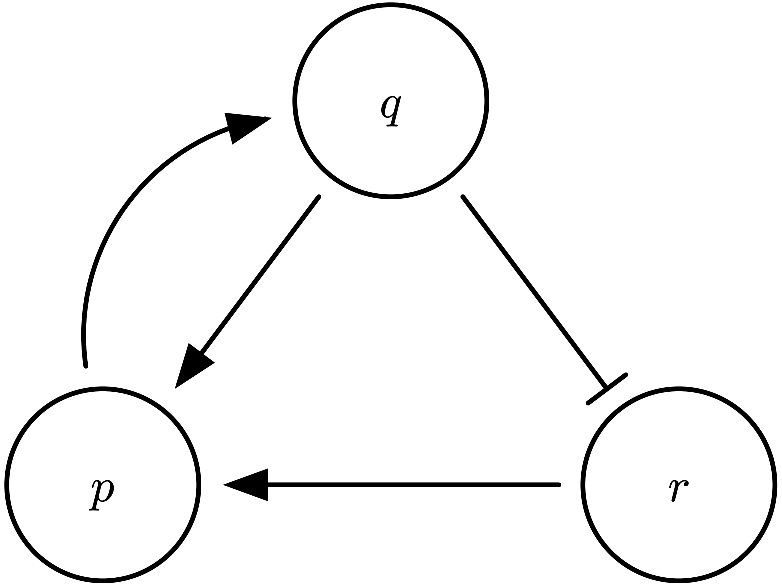
Markov(1) の動的システム

観測できる状態遷移列は

$pqr \to pq \to p \to \varepsilon \to r \to r$

$qr \to pr \to q \to pr$

例



Markov(1) の動的システム

観測できる状態遷移列は

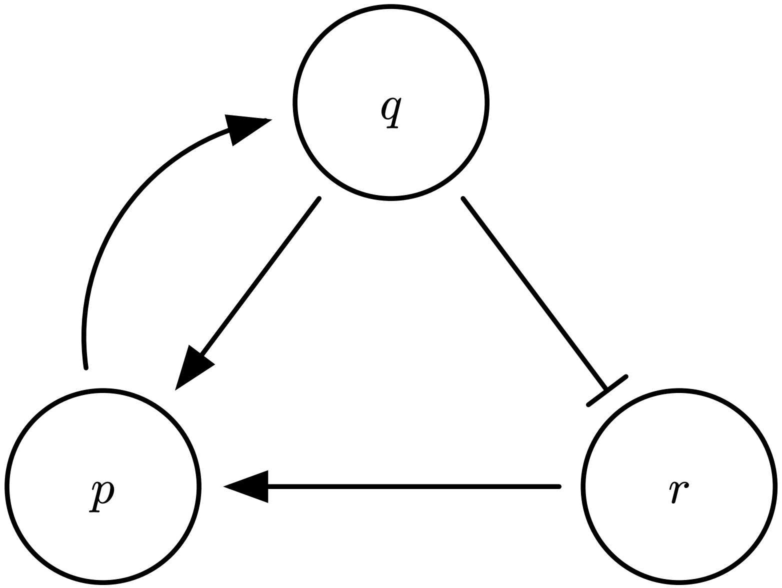
$pqr \to pq \to p \to \varepsilon \to r \to r$

$qr \to pr \to q \to pr$

これを集合に変換

$\{(pqr, pq), (pq, p), (p, \varepsilon), \ldots\}$

例



Markov(1) の動的システム

観測できる状態遷移列は

$pqr \to pq \to p \to \varepsilon \to r \to r$

$qr \to pr \to q \to pr$

これを集合に変換

$\{(pqr, pq), (pq, p), (p, \varepsilon), \ldots\}$

これから学習できる論理プログラムは

$p_{t+1} \leftarrow q_t.$

$q_{t+1} \leftarrow p_t \wedge r_t.$

$r_{t+1} \leftarrow \neg p_t.$

正則形式 (Canonical form)

定理　不変性や対称性を持つ表現では正則形式を定義することができる。

- このような問題において任意の形式から正則形式へと変換する決定的アルゴリズムを定義する
- 深層機械学習モデルに正則形式だけを学習させれば、同じ意味合いを持つ他の形式にも汎化が可能
- しかし、どんな表現でも正則形式を見つけることができるとは限らない
- チューリングマシンにおける停止問題で、形式の異なる二つのプログラムが同等か判断が難しい

標準論理プログラムの正則形式

- P は集合であるため、ルールの順序における不変性質がある。そのため、正則形式のルールの順序を決める
- それぞれのルールにおける正則形式を決める
  - $A \wedge A$ は A に簡略化
  - $A \wedge B$ および $B \wedge A$ は変数のアルファベット順に $A \wedge B$が正則形式
  - $A \wedge \neg A$ は $\perp$
- 正負リテラルはそれぞれ2進数の0と1にみなし、小さい順に並べ替える
- 正則形式でないルールは考慮しない

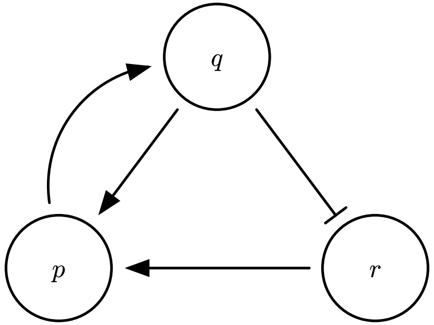
ルールの正則形式および順序

3 変数のルールのみの場合

$l \leq 1$	$l = 2$	$l = 3$
0: $\{\}$	7: $\{a, b\}$	19: $\{a, b, c\}$
1: $\{a\}$	8: $\{a, c\}$	20: $\{\neg a, b, c\}$
2: $\{b\}$	9: $\{b, c\}$	21: $\{a, \neg b, c\}$
3: $\{c\}$	10: $\{\neg a, b\}$	22: $\{\neg a, \neg b, c\}$
4: $\{\neg a\}$	11: $\{\neg a, c\}$	23: $\{a, b, \neg c\}$
5: $\{\neg b\}$	12: $\{\neg b, c\}$	24: $\{\neg a, b, \neg c\}$
6: $\{\neg c\}$	13: $\{a, \neg b\}$	25: $\{a, \neg b, \neg c\}$
	14: $\{a, \neg c\}$	26: $\{\neg a, \neg b, \neg c\}$
	15: $\{b, \neg c\}$	
	16: $\{\neg a, \neg b\}$	
	17: $\{\neg a, \neg c\}$	
	18: $\{\neg b, \neg c\}$	

入力の順不変性

- 状態遷移が入力、Markov(k)の場合$0, ..., k$までの遷移列に順序が決められている
- しかしそれぞれの遷移列間に順序はない
- $0, .., k$までの状態遷移をひとつの要素とみなし、入力は集合
- 集合を入力としたニューラルネットワークモデルが必要
- Set Transformer[8] を利用



観測できる状態遷移列は

$pqr \to pq \to p \to \varepsilon \to r \to r$

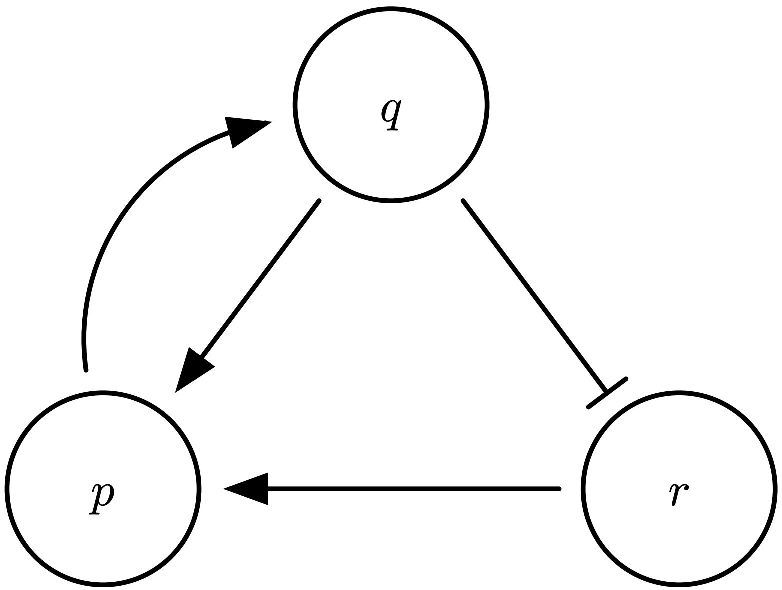
$qr \to pr \to q \to pr$

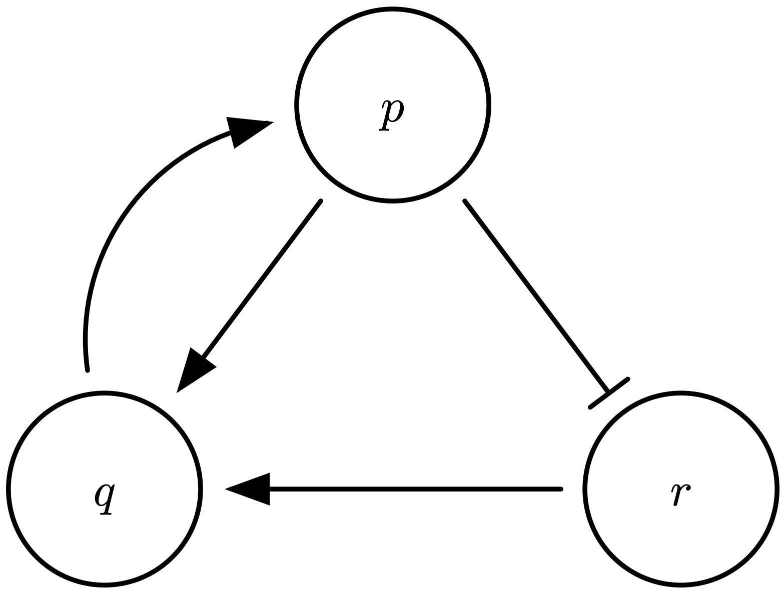
これを集合に変換

$\{(pqr, pq), (pq, p), (p, \varepsilon), (\varepsilon, r), (r, r), (qr, pr), (pr, q), (q, pr)\}$

[8]Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S. and Teh, Y.W., 2019, May. Set transformer: A framework for attention-based permutation-invariant neural networks. In International conference on machine learning (pp. 3744-3753). PMLR.

変数命名による不変性

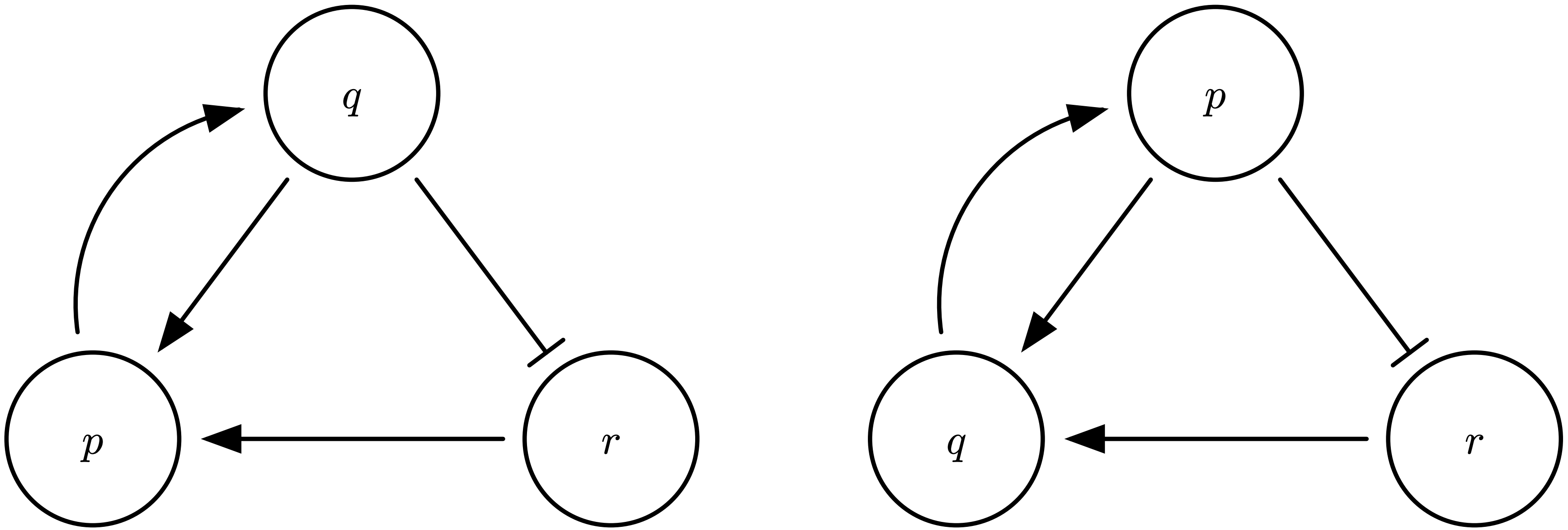




まったく同じ働きをするが、変数の名前だけ異なる

標準論理プログラムの学習

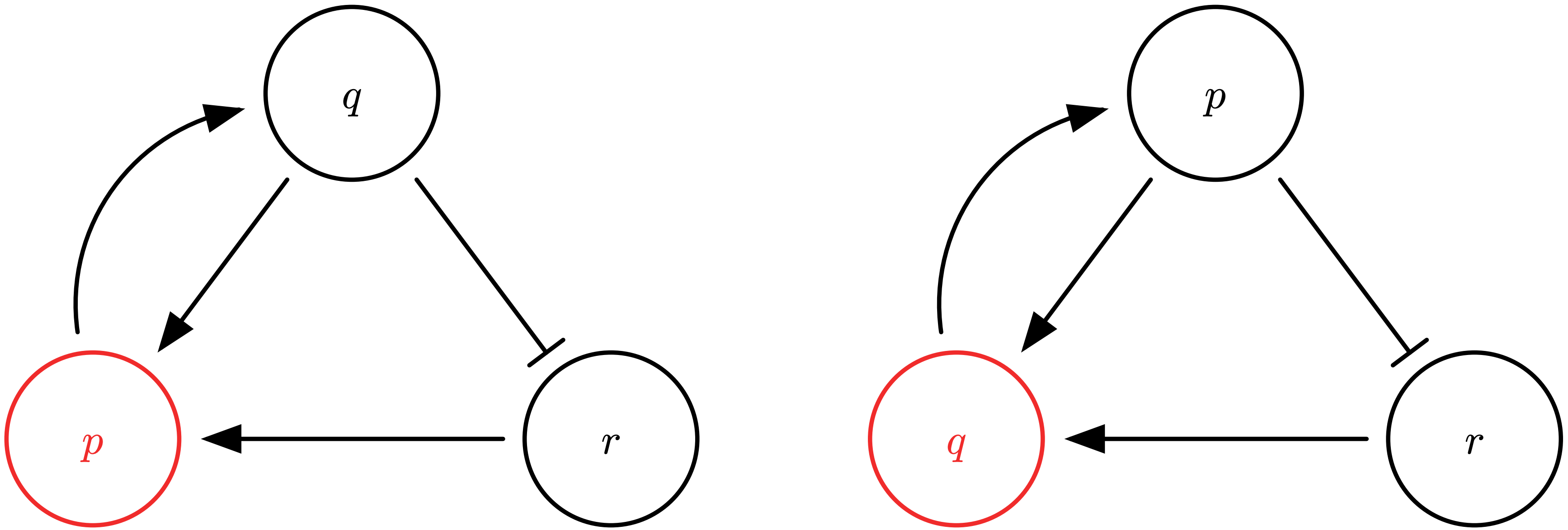
ルールのヘッドがひとつのアトムのみなので、各アトムに分割してルールを学習することが可能



左の p あるいは右の q のルールを学習する際はそれを v_0 と名前を変更する

標準論理プログラムの学習

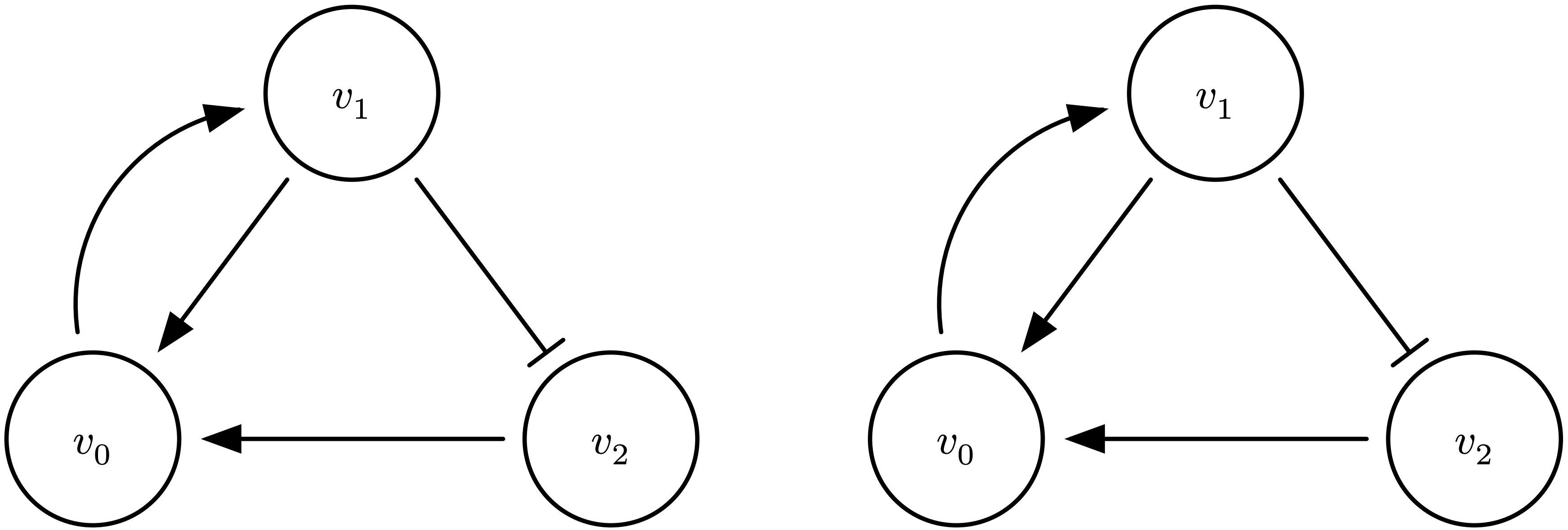
ルールのヘッドがひとつのアトムのみなので、各アトムに分割してルールを学習することが可能



左の p あるいは右の q のルールを学習する際はそれを v_0 と名前を変更する

標準論理プログラムの学習

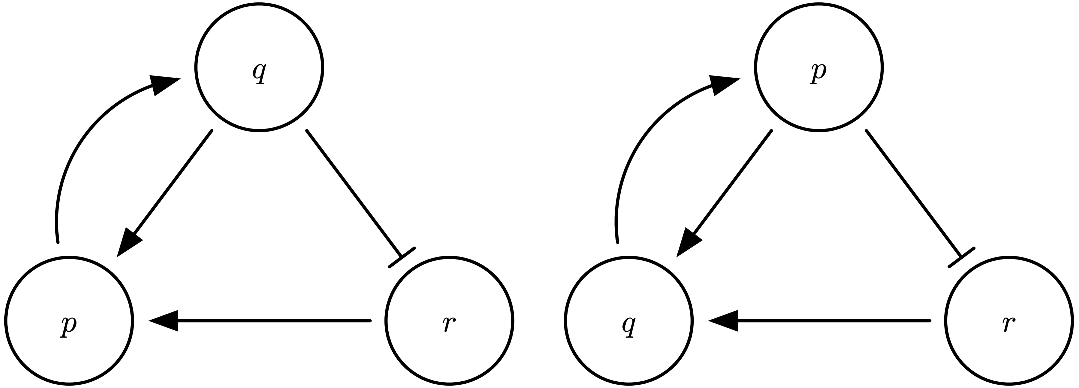
ルールのヘッドがひとつのアトムのみなので、各アトムに分割してルールを学習することが可能



左の p あるいは右の q のルールを学習する際はそれを v_0 と名前を変更する

ニューラルネットワークモデルの視点

p, q, r をベクトル (p, q, r) と表現した場合



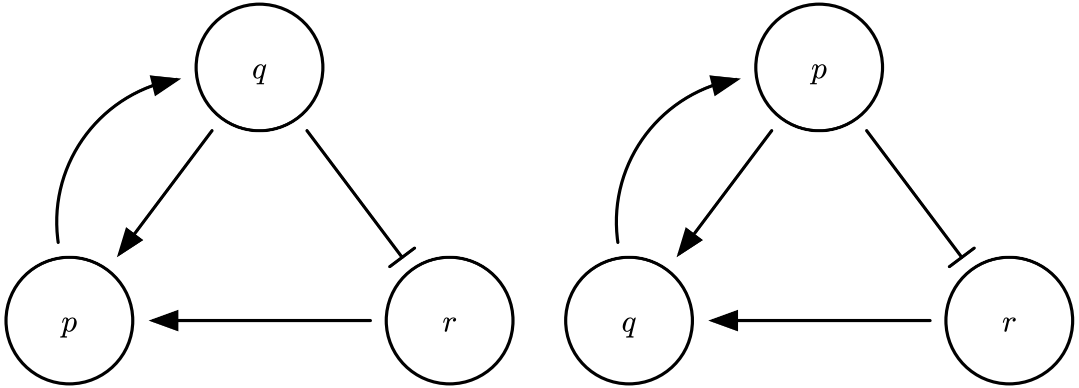
左下のルールはそれぞれ

$p \leftarrow q \wedge r$ (ルール 7)

$q \leftarrow p \wedge r$ (ルール 8)

ニューラルネットワークモデルの視点

p, q, r をベクトル (p, q, r) と表現した場合

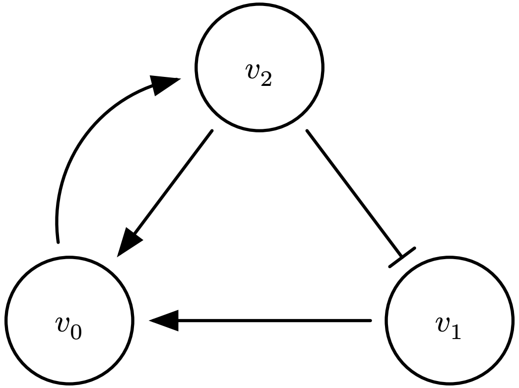


左下のルールはそれぞれ

$p \leftarrow q \wedge r$ (ルール 7)

$q \leftarrow p \wedge r$ (ルール 8)

左下を v_0 とした場合



下記のルールのみを学習すればよい

$v_0 \leftarrow v_1 \wedge v_2$ (ルール 7)

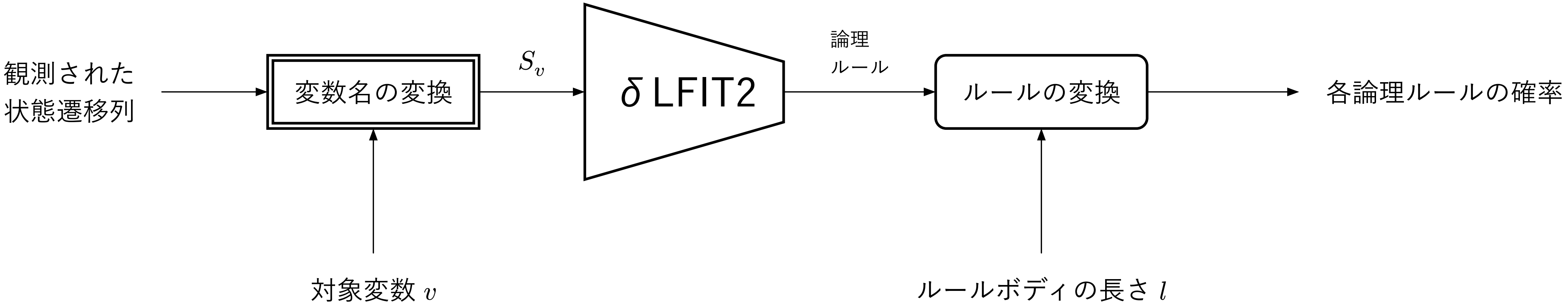
入力空間の削減

不変性を利用することでニューラルネットワークの学習する入力空間を大幅に削減

$pqr \to \{p, q, r, ..., pqr\}$ $pq \to \{p, q, r, ..., pqr\}$ $pr \to \{p, q, r, ..., pqr\}$ $qr \to \{p, q, r, ..., pqr\}$ $p \to \{p, q, r, ..., pqr\}$ $q \to \{p, q, r, ..., pqr\}$ $r \to \{p, q, r, ..., pqr\}$ $\varepsilon \to \{p, q, r, ..., pqr\}$	$v_0v_1v_2 \to \{0, 1\}$ $v_0v_1 \to \{0, 1\}$ $v_0v_2 \to \{0, 1\}$ $v_1v_2 \to \{0, 1\}$ $v_0 \to \{0, 1\}$ $v_1 \to \{0, 1\}$ $v_2 \to \{0, 1\}$ $\varepsilon \to \{0, 1\}$
$(2^n)^{2^n}$	2^{2^n}

δ LFIT2

ニューラルネットワークモデルに入力する前に正則形式に変換し、出力として論理規則を得る。[9]



[9] Phua, Y.J. and Inoue, K., 2024, September. Variable Assignment Invariant Neural Networks for Learning Logic Programs. In International Conference on Neural-Symbolic Learning and Reasoning (pp. 47-61). Cham: Springer Nature Switzerland.

結果

Boolean Network	NN-LFIT	δ LFIT	δ LFIT+	δ LFIT2[3]	δ LFIT2[4]
3-node a (3/2)	**0.000**	0.095	0.271	**0.000**	0.054
3-node b (3/2)	0.042	0.188	0.083	**0.000**	**0.000**
Raf (3/2)	0.333	0.253	0.188	**0.000**	0.025
5-node (5/4)	0.514	**0.142**	0.278	0.238	0.214
7-node (7/3)	0.299	-	0.223	**0.035**	0.152
WNT5A (7/2)	0.063	-	0.194	**0.009**	0.073
Circadian (10/4)	0.260	-	-	**0.033**	0.136
Gene Network (12/3)	0.029	-	-	**0.005**	0.159
Budding Yeast (18/4)	-	-	-	**0.121**	0.307

(n/l): n は動的システム内に含まれている変数、l は論理ルールの最大の長さ

結果

部分遷移列だけ与えられた場合

Given	3-node a			3-node b			Raf		
	NN-LFIT	δLFIT+	δLFIT2	NN-LFIT	δLFIT+	δLFIT2	NN-LFIT	δLFIT+	δLFIT2
3/8	0.542	0.319	**0.308**	**0.292**	0.472	0.362	0.458	**0.319**	0.358
4/8	0.500	**0.264**	0.296	0.403	0.403	**0.321**	0.417	**0.264**	0.287
5/8	0.333	0.389	**0.188**	0.292	0.389	**0.267**	0.458	0.306	**0.242**
6/8	0.375	0.236	**0.183**	0.167	0.222	**0.154**	0.333	0.278	**0.150**
7/8	0.083	0.292	**0.071**	0.083	0.195	**0.079**	0.250	0.236	**0.063**

ニューロシンボリック AI における不変性

- シンボリック表現には対称性が多く存在しているが、それを利用する研究がまだあまりない
- 多くのニューロシンボリック手法はひとつの問題に対してひとつのモデルを学習させているので、不変性の恩恵が少ない
- LLM などの生成 AI によって基盤モデルが注目されている中、ニューロシンボリック AI も基盤モデルが求められている
  - 基盤モデルはより大量のデータから学習するため、より最適なパラメータが見つかると考えられる
- 基盤モデルの構築で不変性を利用することが大変重要である

資料

論文およびコード共に公開している

https://arxiv.org/abs/2408.10709

https://github.com/phuayj/delta-lfit-2

https://yinjunphua.com