UCLA Computer Science

STAR AI
RESEARCH LAB
UCLA

Symbolic Reasoning for Large Language Models

Guy Van den Broeck

Visa Research GenAI Symposium - Sep 18 2024

Outline

1. The paradox of learning to reason from data

   ~~end-to-end learning~~

2. Symbolic reasoning at generation time

3. Symbolic reasoning at training time

   logical + *probabilistic reasoning* + *deep learning*

Outline

1. **The paradox of learning to reason from data**

   ~~*end-to-end learning*~~

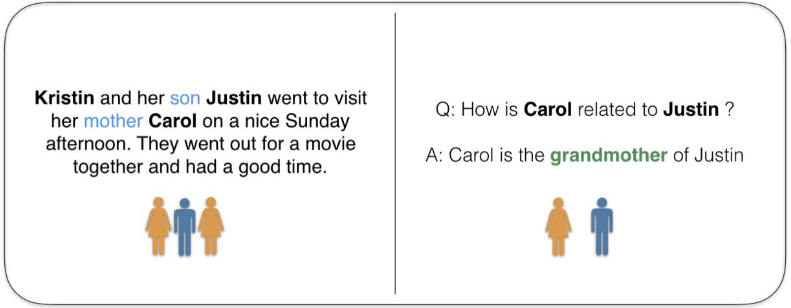
2. Symbolic reasoning at generation time

3. Symbolic reasoning at training time

   logical + probabilistic reasoning + deep learning

Can Language Models Perform Logical Reasoning?

Language Models achieve high performance on “reasoning” benchmarks.



Reasoning Example from the CLUTRR dataset

Unclear whether they follow the rules of logical deduction.

<u>Language Models</u>:
input → ? → Carol is the grandmother of Justin.

<u>Logical Reasoning</u>:
input → Justin in Kristin’s son; Carol is Kristin’s mother; → Carol is Justin’s mother’s mother; if X is Y’s mother’s mother then X is Y’s grandmother → Carol is the grandmother of Justin.

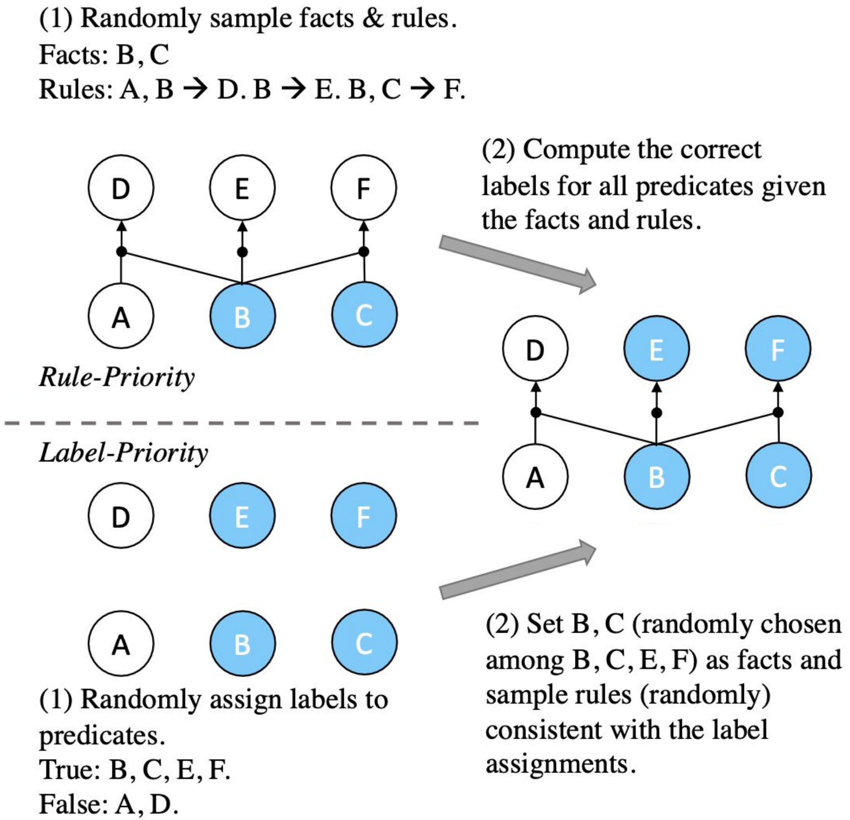
SimpleLogic

Generate textual train and test examples of the form:

Rules: If witty, then diplomatic. If careless and condemned and attractive, then blushing. If dishonest and inquisitive and average, then shy. If average, then stormy. If popular, then blushing. If talented, then hurt. If popular and attractive, then thoughtless. If blushing and shy and stormy, then inquisitive. If adorable, then popular. If cooperative and wrong and stormy, then thoughtless. If popular, then sensible. If cooperative, then wrong. If shy and cooperative, then witty. If polite and shy and thoughtless, then talented. If polite, then condemned. If polite and wrong, then inquisitive. If dishonest and inquisitive, then talented. If blushing and dishonest, then careless. If inquisitive and dishonest, then troubled. If blushing and stormy, then shy. If diplomatic and talented, then careless. If wrong and beautiful, then popular. If ugly and shy and beautiful, then stormy. If shy and inquisitive and attractive, then diplomatic. If witty and beautiful and frightened, then adorable. If diplomatic and cooperative, then sensible. If thoughtless and inquisitive, then diplomatic. If careless and dishonest and troubled, then cooperative. If hurt and witty and troubled, then dishonest. If scared and diplomatic and troubled, then average. If ugly and wrong and careless, then average. If dishonest and scared, then polite. If talented, then dishonest. If condemned, then wrong. If wrong and troubled and blushing, then scared. If attractive and condemned, then frightened. If hurt and condemned and shy, then witty. If cooperative, then attractive. If careless, then polite. If adorable and wrong and careless, then diplomatic. Facts: Alice sensible Alice condemned Alice thoughtless Alice polite Alice scared Alice average
Query: Alice is shy ?

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

Training a transformer on SimpleLogic



Test accuracy for different reasoning depths

Test	0	1	2	3	4	5	6
RP	99.9	99.8	99.7	99.3	98.3	97.5	95.5

Test	0	1	2	3	4	5	6
LP	100.0	100.0	99.9	99.9	99.7	99.7	99.0

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

Has the transformer learned to reason from data?

1. Easiest of reasoning problems (no variance, self-contained, purely symbolic, tractable)
2. RP/LP data covers the whole problem space
3. The learned model has almost 100% test accuracy
4. There exist transformer parameters that compute the ground-truth reasoning function:

Theorem: *For a BERT model with* n *layers and 12 attention heads, by construction, there exists a set of parameters such that the model can correctly solve any reasoning problem in SimpleLogic that requires at most* n − 2 *steps of reasoning.*

Surely, under these conditions, the transformer has learned the ground-truth reasoning function!

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

The Paradox of Learning to Reason from Data

Train	Test	0	1	2	3	4	5	6
RP	RP	99.9	99.8	99.7	99.3	98.3	97.5	95.5
	LP	99.8	99.8	99.3	96.0	90.4	75.0	57.3
LP	RP	97.3	66.9	53.0	54.2	59.5	65.6	69.2
	LP	100.0	100.0	99.9	99.9	99.7	99.7	99.0

The BERT model trained on one distribution fails to generalize to the other distribution within the same problem space.

1. If the transformer **has learned** to reason,
   it should not exhibit such generalization failure.

2. If the transformer **has not learned** to reason,
   it is baffling how it achieves near-perfect in-distribution test accuracy.

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

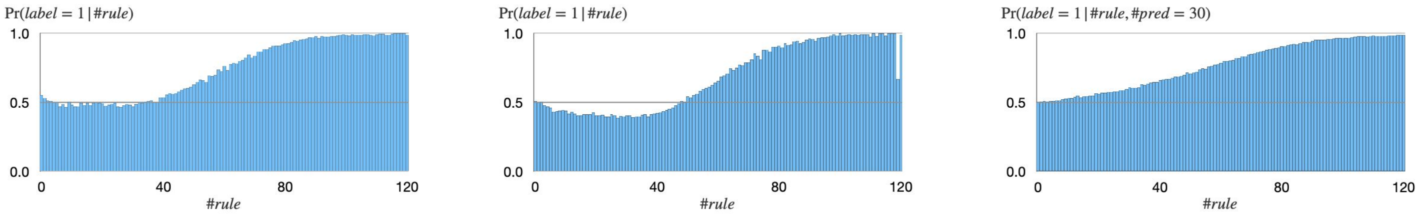
Why? Statistical Features

Monotonicity of entailment:

Any rules can be freely added to the axioms of any proven fact.

The more rules given, the more likely a predicate will be proven.

Pr(label = True | Rule # = x) should increase (roughly) monotonically with x



(a) Statistics for examples generated by Rule-Priority (RP). (b) Statistics for examples generated by Label-Priority (LP). (c) Statistics for examples generated by uniform sampling;

Model leverages statistical features to make predictions

RP_b downsamples from RP such that Pr(label = True | rule# = x) = 0.5 for all x

Train	Test	0	1	2	3	4	5	6
	RP	99.9	99.8	99.7	99.3	98.3	97.5	95.5
RP	RP_b	99.0	99.3	98.5	97.5	96.7	93.5	88.3

1. Accuracy drop from RP to RP_b indicates that
   the model is using rule# as a statistical feature to make predictions.

2. Potentially countless statistical features

3. Such features are **inherent to the reasoning problem**, cannot make data "clean"

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

First Conclusion

Experiments unveil the fundamental difference between

1. learning to reason, and
2. learning to achieve high performance on benchmarks using statistical features.

Be careful deploying AI in applications where this difference matters.

FAQ: Do bigger transformers solve this problem? No, already 99% accurate…

FAQ: Will reasoning emerge? Perhaps on 99% of predictable human behavior…

We won't invent jazz or calculus that way…

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang and Guy Van den Broeck. On the Paradox of Learning to Reason from Data, 2022

Current investigation…

4. There exist transformer parameters that compute the ground-truth reasoning function:

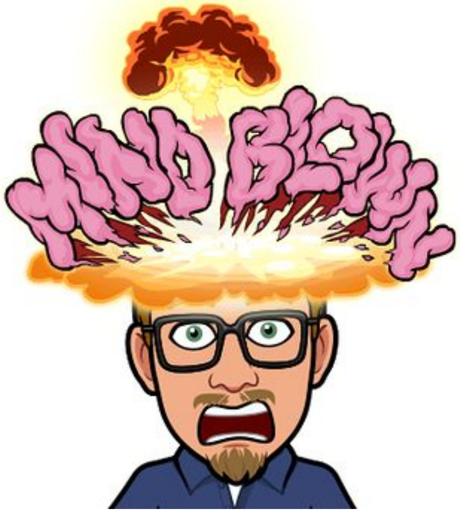
   Theorem 1: *For a BERT model with n layers and 12 attention heads, by construction, there exists a set of parameters such that the model can correctly solve any reasoning problem in SimpleLogic that requires at most n − 2 steps of reasoning.*

Initialize with the perfect parameters

that simulate the ground-truth reasoning algorithm.

Then SGD will **un-learn the algorithm** that generalizes OOD?

… we don’t understand what is going on …



Outline

1. The paradox of learning to reason from data

   ~~end-to-end learning~~

2. **Symbolic reasoning at generation time**

3. Symbolic reasoning at training time

   logical + *probabilistic reasoning* + *deep learning*

Generate a sentence using "frisbee", "caught" and "dog", following the given order.

Generate a sentence using "frisbee", "caught" and "dog", following the given order.

After a perfect throw, the frisbee glided through the air, and the dog, with incredible agility, caught it mid-flight.

ChatGPT

Generate a sentence using "frisbee", "caught" and "dog", following the given order.

After a perfect throw, the frisbee glided through the air, and the dog, with incredible agility, caught it mid-flight.

ChatGPT

That's not correct. Generate a sentence using "frisbee", "caught" and "dog". The keywords should appear in the order as specified.

Here's the correct sentence: The dog caught the frisbee in mid-air, showing off its amazing catching skills.

ChatGPT

A frisbee is caught by a dog.

A pair of frisbee players are caught in a dog fight.

Ctrl-G

What do we have?

Prefix: "The weather is"

Constraint α: text contains "winter"

Model only does $p(\text{next-token}|\text{prefix}) =$

cold	0.05
warm	0.10

Train some $q(.\,|\alpha)$ for a specific task distribution $\alpha \sim p_{\text{task}}$

Train $q(\text{next-token}|\text{prefix}, \alpha)$ and avoid symbolic reasoning

BEWARE OF THE PARADOX

What do we need?

Prefix: “The weather is”

Constraint α: text contains “winter”

Generate from $p(\text{next-token}|\text{prefix}, \alpha) =$

cold	0.50
warm	0.01

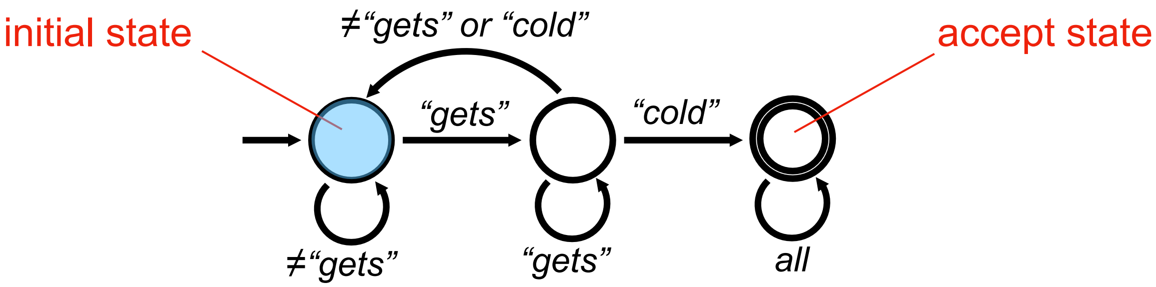
$$\propto \sum_{\text{text}} p(\text{next-token}, \text{text}, \text{prefix}, \alpha)$$

Marginalization! Probabilistic Reasoning!

Representing Logical Constraints as DFAs

A deterministic finite automaton (DFA) checks whether a string satisfies certain constraints.

Example. Check if a string contains “gets cold”.

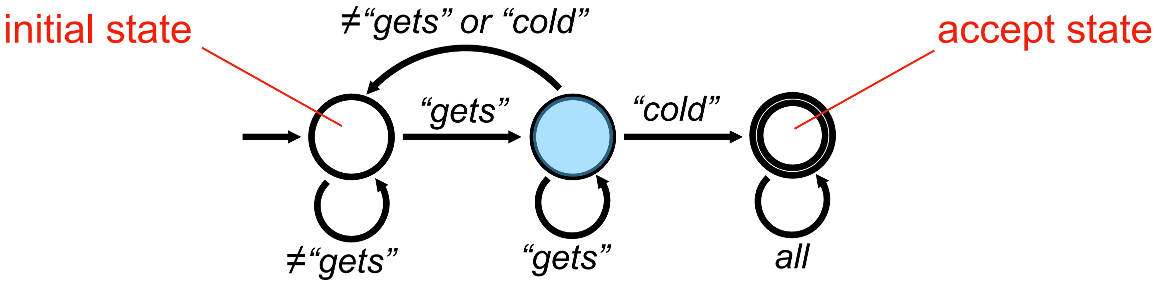


String: “The weather gets cold in the winter.”

Representing Logical Constraints as DFAs

A deterministic finite automaton (DFA) checks whether a string satisfies certain constraints.

Example. Check if a string contains “gets cold”.

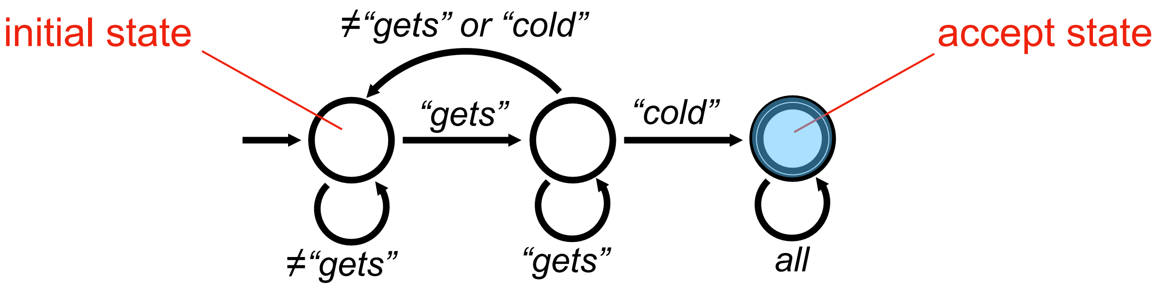


String: “The weather gets cold in the winter.”

Representing Logical Constraints as DFAs

A deterministic finite automaton (DFA) checks whether a string satisfies certain constraints.

Example. Check if a string contains “gets cold”.



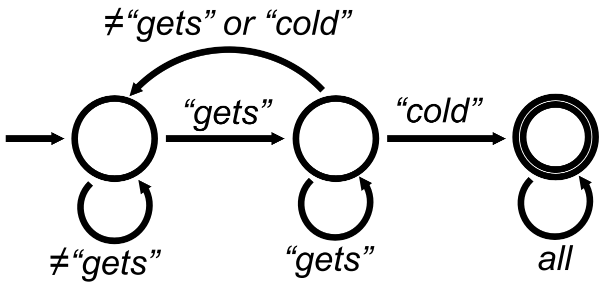
String: “The weather gets cold in the winter.”

Representing Logical Constraints as DFAs

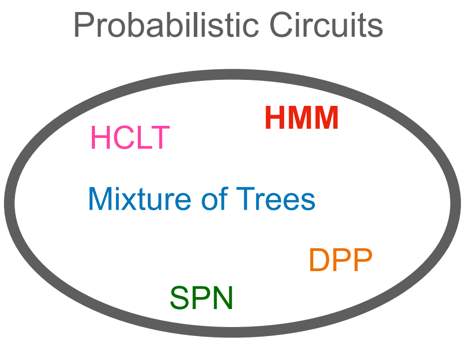
A deterministic finite automaton (DFA) checks whether a string satisfies certain constraints.

Can represent:

1. Phrases/words must/must not appear
2. Exactly k words/sentences/paragraphs.
3. Only words from a given vocabulary.
4. String must end a certain way
5. Any regex
6. Anything over fixed sequence lengths
7. …

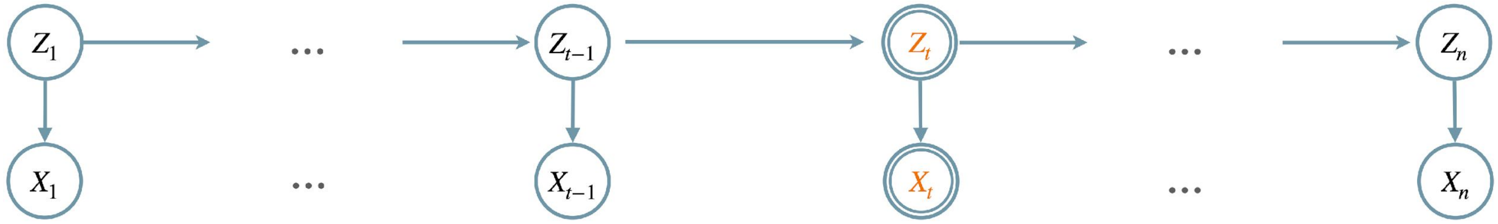


Tractable Deep Generative Models



Model joint probability distributions *and* allow efficient probabilistic inference

Keep it simple… just a classic **Hidden Markov Model** (HMM) with 32,768 hidden states and 2 billion parameters… on the GPU



***Theorem**. Given a DFA constraint α with m edges and an HMM p(x) with h hidden states, computing $p(\alpha \mid x_{1:t+1})$ over a sequence of n tokens takes $O(nmh^2)$ time.*

Honghua Zhang, Po-Nien Kung, Masahiro Yoshida, Guy Van den Broeck and Nanyun Peng. Adaptable Logical Control for Large Language Models, *In Arxiv*, 2024.

The Ctrl-G Architecture

Lexical Constraint α: sentence contains keyword "winter"

Constrained Generation: $\Pr(x_{t+1} \mid \alpha, x_{1:t} =$ "the weather is"$)$

✗ intractable → Pre-trained Language Model

✓ efficient → Tractable Probabilistic Model

Minimize KL-divergence

x_{t+1}	$\Pr_{LM}(x_{t+1} \mid x_{1:t})$
cold	0.05
warm	0.10

x_{t+1}	$\Pr_{TPM}(\alpha \mid x_{t+1}, x_{1:t})$
cold	0.50
warm	0.01

The Ctrl-G Architecture

Lexical Constraint α: sentence contains keyword "winter"

Constrained Generation: $\Pr(x_{t+1} \mid \alpha, x_{1:t} = \text{"the weather is"})$

intractable → Pre-trained Language Model

efficient → Tractable Probabilistic Model

Minimize KL-divergence

x_{t+1}	$\Pr_{LM}(x_{t+1} \mid x_{1:t})$
cold	0.05
warm	0.10

x_{t+1}	$\Pr_{TPM}(\alpha \mid x_{t+1}, x_{1:t})$
cold	0.50
warm	0.01

x_{t+1}	$p(x_{t+1} \mid \alpha, x_{1:t})$
cold	0.025
warm	0.001

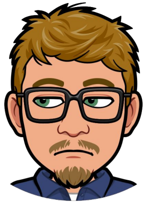
By Bayes rule,

p_{LM}(next-token | α, prefix)

$\propto$

p_{LM}(next-token | prefix)

$\cdot$ p_{LM}(α | next-token, prefix)



The Ctrl-G Architecture

Lexical Constraint α: sentence contains keyword "winter"

Constrained Generation: $\Pr(x_{t+1} \mid \alpha, x_{1:t} =$ "the weather is"$)$

✗ intractable → Pre-trained Language Model

✓ efficient → Tractable Probabilistic Model

Tractable Probabilistic Model → Pre-trained Language Model: Minimize KL-divergence

x_{t+1}	$\Pr_{LM}(x_{t+1} \mid x_{1:t})$
cold	0.05
warm	0.10

x_{t+1}	$\Pr_{TPM}(\alpha \mid x_{t+1}, x_{1:t})$
cold	0.50
warm	0.01

x_{t+1}	$p(x_{t+1} \mid \alpha, x_{1:t})$
cold	0.025
warm	0.001

By Bayes rule,

$$p_{CTRL\text{-}G}(\text{next-token} \mid \alpha, \text{prefix}) \propto p_{LM}(\text{next-token} \mid \text{prefix}) \cdot p_{TPM}(\alpha \mid \text{next-token}, \text{prefix})$$

CommonGen Benchmark

Generate a sentence using 3 to 5 concepts (keywords).

Input: snow drive car

α = ("car" ∨ "cars"…) ∧ ("drive" ∨ "drove"…) ∧

Reference 1: A car drives down a snow-covered road.

Reference 2: Two cars drove through the snow.

	BLEU-4		ROUGE-L		CIDEr		SPICE		Constraint	
	dev	*test*	*dev*	*test*	*dev*	*test*	*dev*	*test*	*dev*	*test*
supervised - base models trained with full supervision										
FUDGE	-	24.6	-	40.4	-	-	-	-	-	47.0%
A*esque	-	28.2	-	43.4	-	15.2	-	30.8	-	98.8%
NADO	30.8	-	44.4	-	16.1	-	32.0	-	88.8%	-
Ctrl-G	**35.1**	**34.4**	**46.7**	**46.4**	**17.4**	**17.6**	**32.7**	**33.3**	**100.0%**	**100.0%**
unsupervised - base models not trained with keywords as supervision										
A*esque	-	28.6	-	44.3	-	15.6	-	29.6	-	-
NADO	26.2	-	-	-	-	-	-	-	-	-
Ctrl-G	**32.1**	**31.5**	**45.2**	**44.8**	**16.0**	**16.2**	**30.8**	**31.2**	**100.0%**	**100.0%**

Honghua Zhang, Po-Nien Kung, Masahiro Yoshida, Guy Van den Broeck and Nanyun Peng. Adaptable Logical Control for Large Language Models, *In Arxiv*, 2024.

Interactive Text Editing

CoAuthor

An Open-Source Interface for Human-Language Model (LM) Interaction

User: given the following context, generate infilling text for [BLANK] using key phrases "alien mothership", "far from over"; generated text must contain 25 - 30 words.

"First they've defeated a small squad [BLANK] are few humans left, and despite their magical power, their numbers are getting fewer."

5 lines of code!

```
from CtrlG import *

prefix = "First they defeated a …"
suffix = "are few humans left …"

dfa_list = [
  DFA_all_of("alien mothership",
             "far from over"),
  DFA_word_count(25, 30),
]
dfa = DFA_logical_and(dfa_list)

lp = CtrlGLogitsProcessor(
       dfa, hmm, prefix, suffix)
llm.generate(logits_processor=lp)
```

"First they've defeated a small squad of aliens, then a larger fleet of their ships. Eventually they've even managed to take down the <u>alien mothership</u>. But their problems are <u>far from over</u>. There are few humans left, and despite their magical power, their numbers are getting fewer."

Honghua Zhang, Po-Nien Kung, Masahiro Yoshida, Guy Van den Broeck and Nanyun Peng. Adaptable Logical Control for Large Language Models, *In Arxiv*, 2024.

Interactive Text Editing

CoAuthor

An Open-Source Interface for Human-Language Model (LM) Interaction

	Insertion			
	None	*K*	*L*	*K&L*
Quality				
TULU2	2.68	2.64	2.78	2.74
GPT3.5	2.27	2.22	2.27	2.31
GPT4	**3.79**	3.33	3.53	3.10
Ctrl-G	**3.77**	**3.56**	**3.73**	**3.59**
Success				
TULU2	-	12%	20%	3%
GPT3.5	-	22%	54%	10%
GPT4	-	60%	20%	27%
Ctrl-G	-	**100%**	**100%**	**100%**
Overall				
TULU2	-	7%	10%	1%
GPT3.5	-	0%	5%	2%
GPT4	-	41%	17%	14%
Ctrl-G	-	**76%**	**78%**	**82%**

→ Insert with key phrase (K) or length (L) constraints

→ Ask humans to assign quality scores (out of 5)

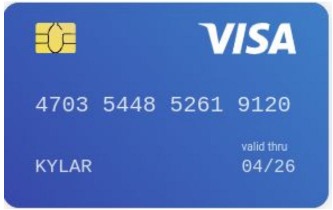
→ Does the output satisfy the constraints?

→ How often does the output satisfy the constraints and achieve a quality above 3?

→ **Ctrl-G based on TULU2-7B wipes the floor with GPT4, which is a >100x bigger LLM**

Honghua Zhang, Po-Nien Kung, Masahiro Yoshida, Guy Van den Broeck and Nanyun Peng. Adaptable Logical Control for Large Language Models, *In Arxiv*, 2024.

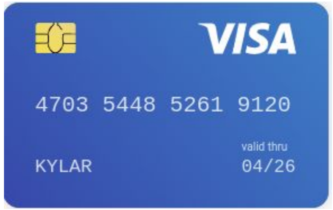
Grade School Math Benchmark



Question: *Kylar went to the store to buy glasses for his new apartment. One glass costs $5, but every second glass costs only 60% of the price. Kylar wants to buy 16 glasses. How much does he need to pay for them?*

Vanilla LLM Answer: The price of the 2nd glass is (16 / 2) * 60% = 8 dollars. So one pair of glasses costs 16 + 8 = 24 dollars. So the answer is 24.

Grade School Math Benchmark



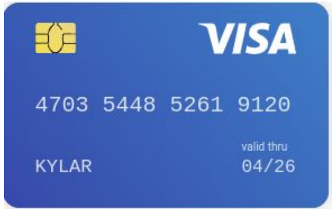
Question: *Kylar went to the store to buy glasses for his new apartment. One glass costs $5, but every second glass costs only 60% of the price. Kylar wants to buy 16 glasses. How much does he need to pay for them?*

Vanilla LLM Answer: The price of the 2nd glass is (16 / 2) * 60% = 8 dollars. So one pair of glasses costs 16 + 8 = 24 dollars. So the answer is 24.

Ctrl-G Answer: The second glass costs 5 * .6 = $3. So each set of two glasses actually costs 5 + 3 = $8. He wants 16 / 2 = 8 sets of two. That means he needs to pay 8 * 8 = $64. So the answer is 64.

Which constraint improves accuracy?

Grade School Math Benchmark



***Question:** Kylar went to the store to buy glasses for his new apartment. One glass costs $**5**, but every second glass costs only **60%** of the price. Kylar wants to buy **16** glasses. How much does he need to pay for them?*

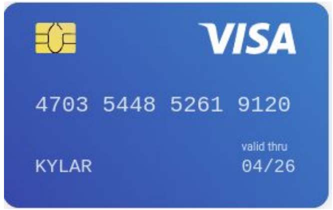
Vanilla LLM Answer: The price of the 2nd glass is (16 / 2) * **60%** = 8 dollars. So one pair of glasses costs **16** + 8 = 24 dollars. So the answer is 24.

Ctrl-G Answer: The second glass costs **5** * **.6** = $3. So each set of two glasses actually costs **5** + 3 = $8. He wants **16** / 2 = 8 sets of two. That means he needs to pay 8 * 8 = $64. So the answer is 64.

Which constraint improves accuracy?

Honghua Zhang, Po-Nien Kung, Masahiro Yoshida, Guy Van den Broeck and Nanyun Peng. Adaptable Logical Control for Large Language Models, *In Arxiv*, 2024.

Grade School Math Benchmark



Question: *Kylar went to the store to buy glasses for his new apartment. One glass costs $**5**, but every second glass costs only **60%** of the price. Kylar wants to buy **16** glasses. How much does he need to pay for them?*

Vanilla LLM Answer: The price of the 2nd glass is (16 / 2) * **60%** = 8 dollars. So one pair of glasses costs **16** + 8 = 24 dollars. So the answer is 24.

Ctrl-G Answer: The second glass costs **5** * **.6** = $3. So each set of two glasses actually costs **5** + 3 = $8. He wants **16** / 2 = 8 sets of two. That means he needs to pay 8 * 8 = $64. So the answer is 64.

Use all the numbers in the problem statement!

Honghua Zhang, Po-Nien Kung, Masahiro Yoshida, Guy Van den Broeck and Nanyun Peng. Adaptable Logical Control for Large Language Models, *In Arxiv*, 2024.

Advantages of Ctrl-G:

1. Constraint α is guaranteed to be satisfied:
   for any next-token x_{t+1} that would make α unsatisfiable, $p(x_{t+1} \mid x_{1:t}, \alpha) = 0$.

2. Training the tractable deep generative model does not depend on α,
   which is only imposed at inference (generation) time.

Conclusion:

You can control an intractable generative model using a generative model that is *tractable for reasoning*.

Outline

1. The paradox of learning to reason from data

   ~~*end-to-end learning*~~

2. Symbolic reasoning at generation time

3. **Symbolic reasoning at training time**

   logical + *probabilistic reasoning* + *deep learning*

Neurosymbolic learning of transformers

Given:

1. constraint α (a list of 403 toxic words not to say)
2. training data D

Learn: a transformer Pr(.) that

1. satisfies the constraint α: $\Pr(\alpha)\uparrow$
2. maximizes the likelihood: $\Pr(D)\uparrow$

Kareem Ahmed, Kai-Wei Chang and Guy Van den Broeck. A Pseudo-Semantic Loss for Deep Generative Models with Logical Constraints, *In Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

Neurosymbolic learning of transformers

Given:

1. constraint α (a list of 403 toxic words not to say)
2. training data D

Learn: a transformer Pr(.) that

1. satisfies the constraint α: Pr(α)↑
2. maximizes the likelihood: Pr(D)↑

Pr(α) is computationally hard, even when α is trivial:

What is probability that LLM ends the sentence with "UCLA"?

Kareem Ahmed, Kai-Wei Chang and Guy Van den Broeck. A Pseudo-Semantic Loss for Deep Generative Models with Logical Constraints, *In Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

Autoregressive distributions are hard…

Pr(α) is computationally hard, even when α is trivial:

What is prob. that LLM ends the sentence with "UCLA"?

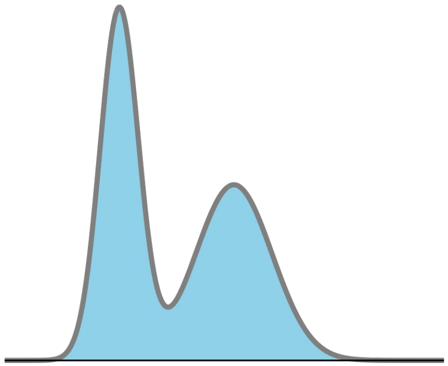
Why did it work before?

We were using a separate **tractable proxy** model…

Now we need to train the actual intractable transformer…

Kareem Ahmed, Kai-Wei Chang and Guy Van den Broeck. A Pseudo-Semantic Loss for Deep Generative Models with Logical Constraints, *In Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

Neuro-Symbolic AI: A Probabilistic Perspective

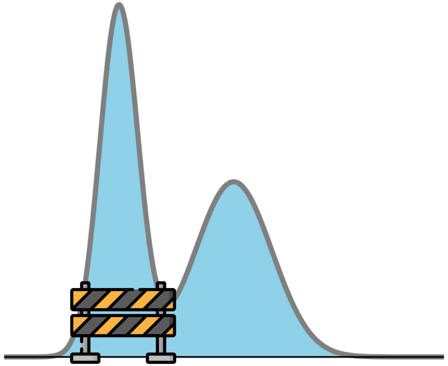


A neural network
induces a distribution

[Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang and Guy Van den Broeck. A Semantic Loss Function for Deep Learning with Symbolic Knowledge, *ICML*, 2018]

Neuro-Symbolic AI: A Probabilistic Perspective

Impose structure using symbolic knowledge

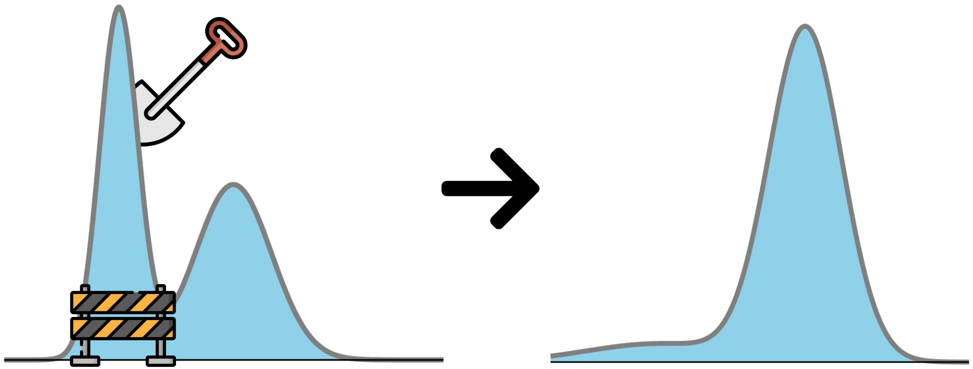


A neural network induces a distribution

[Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang and Guy Van den Broeck. A Semantic Loss Function for Deep Learning with Symbolic Knowledge, *ICML*, 2018]

Neuro-Symbolic AI: A Probabilistic Perspective

Impose structure using symbolic knowledge



A neural network induces a distribution

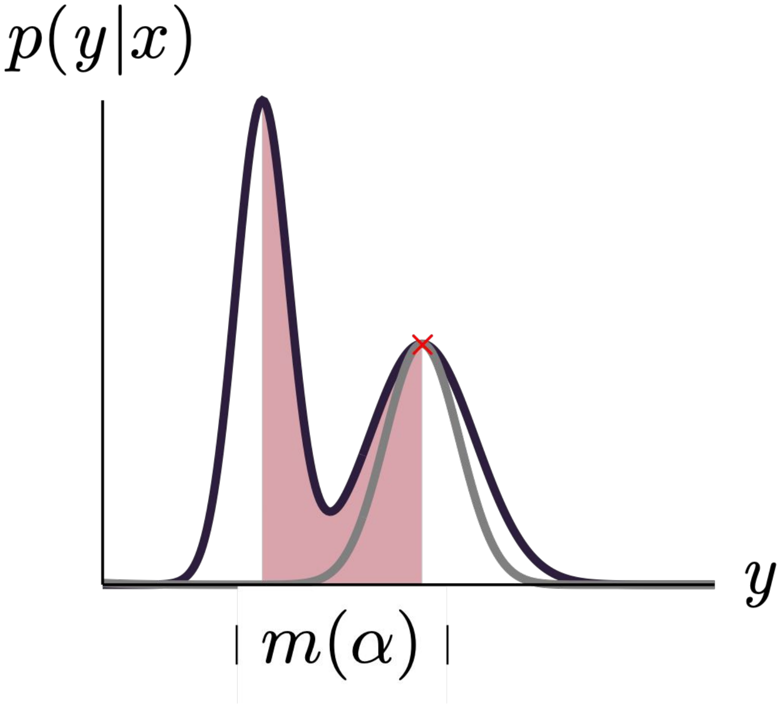
Move mass around to be consistent with structure

[Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang and Guy Van den Broeck. A Semantic Loss Function for Deep Learning with Symbolic Knowledge, *ICML*, 2018]

Neurosymbolic learning of transformers

Basic Idea:

Use how likely a constraint is to be satisfied around a model sample (x) as a proxy for how likely it is to be satisfied under the entire distribution. Average over many such samples.



Kareem Ahmed, Kai-Wei Chang and Guy Van den Broeck. A Pseudo-Semantic Loss for Deep Generative Models with Logical Constraints, *In Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

Formally, minimize the *pseudo-semantic loss*

$$\mathcal{L}^{\mathsf{SL}}_{\mathsf{pseudo}} := -\log \mathbb{E}_{\tilde{\boldsymbol{y}} \sim p} \sum_{\boldsymbol{y} \models \alpha} \prod_{i=1}^{n} p(\boldsymbol{y}_i \mid \tilde{\boldsymbol{y}}_{-i})$$

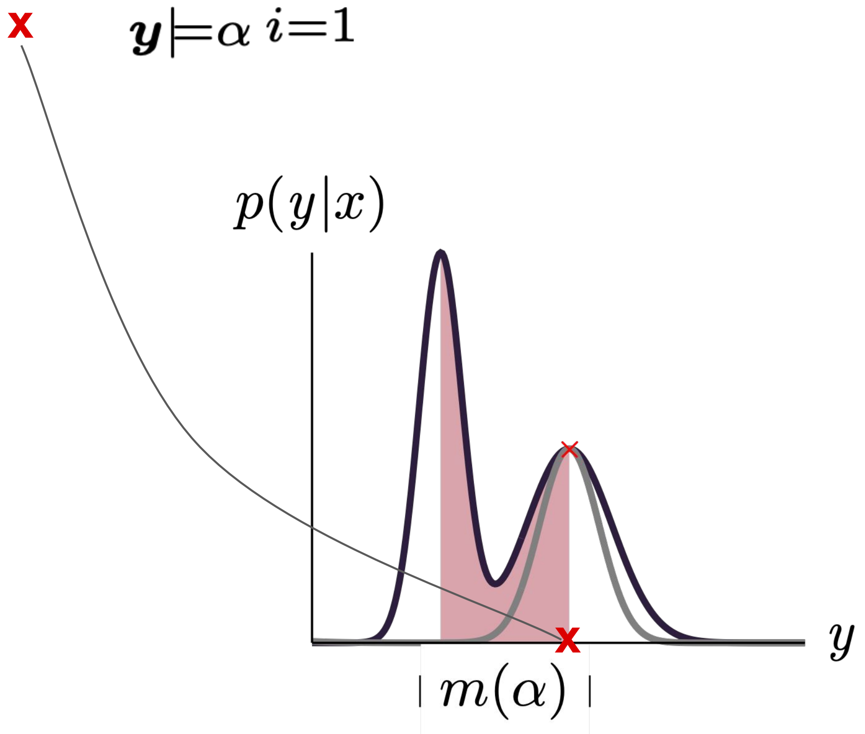
Kareem Ahmed, Kai-Wei Chang and Guy Van den Broeck. A Pseudo-Semantic Loss for Deep Generative Models with Logical Constraints, *In Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

Formally, minimize the *pseudo-semantic loss*

$$\mathcal{L}^{\text{SL}}_{\text{pseudo}} := -\log \mathbb{E}_{\tilde{\boldsymbol{y}} \sim p} \sum_{\boldsymbol{y} \models \alpha} \prod_{i=1}^{n} p(\boldsymbol{y}_i \mid \tilde{\boldsymbol{y}}_{-i})$$

Basic Idea:

Pick a location to build the approximation around

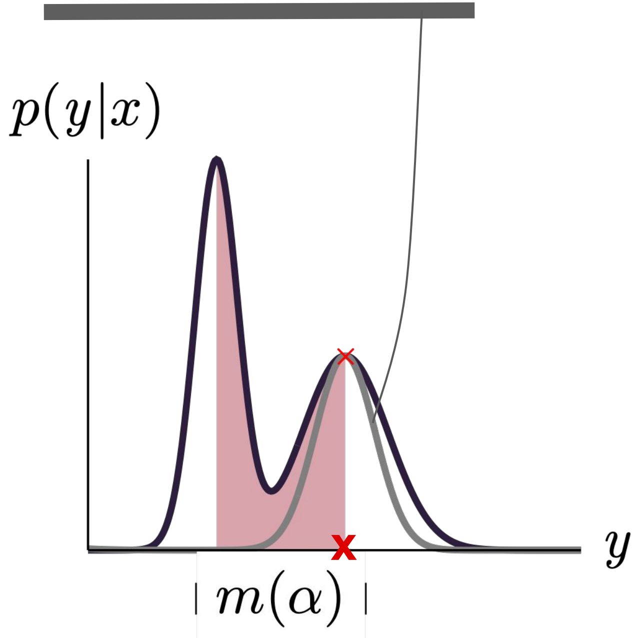


Formally, minimize the *pseudo-semantic loss*

$$\mathcal{L}^{\text{SL}}_{\text{pseudo}} := -\log \mathbb{E}_{\tilde{\boldsymbol{y}} \sim p} \sum_{\boldsymbol{y} \models \alpha} \prod_{i=1}^{n} p(\boldsymbol{y}_i \mid \tilde{\boldsymbol{y}}_{-i})$$

Basic Idea:

Extract a local tractable probabilistic model around the point (independent in each dimension)



How to compute pseudo-semantic loss?

Transformer output gives all alternative next-token logits for ỹ:

$$\begin{array}{lcccccccccc} & p(\text{She}) & & p(\text{caught}|\text{I}) & & p(\text{the}|\text{I, saw}) & & p(\text{cat}|\text{I, saw, a}) & & p(\text{yesterday}|\text{I, saw, a, dog}) \\ p(\text{I saw a dog today}) = & p(\text{I}) & \times & p(\text{saw}|\text{I}) & \times & p(\text{a}|\text{I, saw}) & \times & p(\text{dog}|\text{I, saw, a}) & \times & p(\text{today}|\text{I, saw, a, dog}) \\ & p(\text{He}) & & p(\text{bought}|\text{I}) & & p(\text{an}|\text{I, saw}) & & p(\text{mouse}|\text{I, saw, a}) & & p(\text{tomorrow}|\text{I, saw, a, dog}) \\ & \cdots & & \cdots & & \cdots & & \cdots & & \cdots \end{array}$$

Just reuse these probabilities

$$p(\text{I saw a mouse today}) = p(\text{I}) \times p(\text{saw}|\text{I}) \times p(\text{a}|\text{I, saw}) \times p(\text{mouse}|\text{I, saw, a}) \times p(\text{today}|\text{I, saw, a, dog})$$

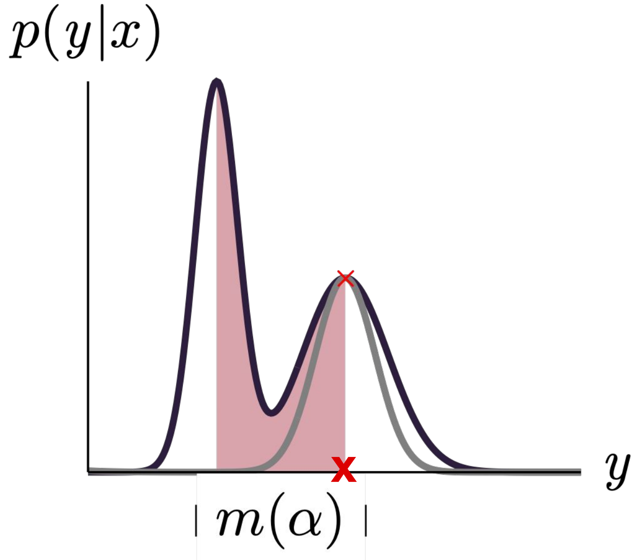
Kareem Ahmed, Kai-Wei Chang and Guy Van den Broeck. A Pseudo-Semantic Loss for Deep Generative Models with Logical Constraints, *In Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

Formally, minimize the *pseudo-semantic loss*

$$\mathcal{L}^{\mathsf{SL}}_{\mathsf{pseudo}} := -\log \mathbb{E}_{\tilde{\boldsymbol{y}} \sim p} \sum_{\boldsymbol{y} \models \alpha} \prod_{i=1}^{n} p(\boldsymbol{y}_i \mid \tilde{\boldsymbol{y}}_{-i})$$

Basic Idea:

Compute Pr(α) locally and maximize it

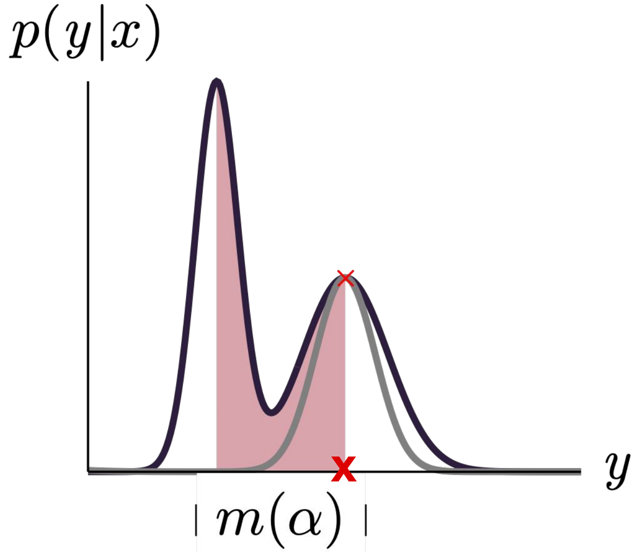


Formally, minimize the *pseudo-semantic loss*

$$\mathcal{L}^{\mathsf{SL}}_{\mathsf{pseudo}} := -\log \mathbb{E}_{\tilde{\boldsymbol{y}} \sim p} \sum_{\boldsymbol{y} \models \alpha} \prod_{i=1}^{n} p(\boldsymbol{y}_i \mid \tilde{\boldsymbol{y}}_{-i})$$

How good is this approximation?

- **Local:**
  ~30 bits entropy vs ~80 for GPT-2.
- **Fidelity:**
  4 bits KL-divergence from GPT-2.



Detoxify LLMs by disallowing bad words

Constraint α is a list of 403 toxic words not to say
Evaluation is a toxicity classifier

Models		Exp. Max. Toxicity (↓)			Toxicity Prob. (↓)			PPL (↓)
		Full	Toxic	Nontoxic	Full	Toxic	Nontoxic	
	GPT-2	0.44	0.62	0.39	34.11%	67.27%	24.85%	25.85
Domain-Adaptive	SGEAT [42]	0.32	0.46	0.28	14.05%	35.72%	7.99%	28.72
	PseudoSL *(ours)*	**0.29**	0.38	**0.27**	9.80%	20.07%	6.93%	28.14
Word Banning	GPT-2	0.40	0.55	0.36	27.92%	57.86%	19.56%	22.24
	SGEAT [42]	0.30	0.41	**0.27**	10.73%	27.05%	**6.17%**	24.91
	PseudoSL *(ours)*	**0.29**	**0.37**	**0.27**	**9.20%**	**18.71%**	6.55%	24.19

Kareem Ahmed, Kai-Wei Chang and Guy Van den Broeck. A Pseudo-Semantic Loss for Deep Generative Models with Logical Constraints, *In Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

Outline

1. The paradox of learning to reason from data

   ~~end-to-end learning~~

2. Symbolic reasoning at generation time

3. Symbolic reasoning at training time

   logical + *probabilistic reasoning* + *deep learning*

Thanks

This was the work of many wonderful students/postdocs/collaborators!

References: http://starai.cs.ucla.edu/publications/