Augmenting Neural Networks with First-order Logic

Tao Li[1], **Vivek Srikumar**[1]

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019.

[1]University of Utah.

Juliete Rossie, 01/30/2026

Introduction

context: **2019 vs 2026**

Paper example

Paragraph:
Gaius Julius Caesar (July 100 BC –15 March 44 BC), **Roman general**, statesman, Consul and notable **author** of **Latin prose**, played a critical role in the events that led to the demise of the Roman Republic and the rise of the Roman Empire through his various military campaigns.

Question:
Which **Roman general** is known for **writing** **prose**?

Background

Background
Propositional vs First-order Logic

Propositional Logic

Based on propositions and relationships between them

Propositions: p, q, r, ...

Relationships:

AND	$p \wedge q$
OR	$p \vee q$
NOT	$\neg p$
implies	$p \Rightarrow q$
...	...

Can you represent this example in Propositional logic?

- Alice is taller than Bob
- Bob is taller than Charlie
- Therefore Alice is taller than Charlie

First-order Logic

Background
Propositional vs First-order Logic

Propositional Logic

Based on propositions and relationships between them

Propositions: p, q, r, ...

Relationships:

AND	$p \wedge q$
OR	$p \vee q$
NOT	$\neg p$
implies	$p \Rightarrow q$
...	...

Can you represent this example in Propositional logic?

- Alice is taller than Bob
- Bob is taller than Charlie
- Therefore Alice is taller than Charlie

First-order Logic

Background
Propositional vs First-order Logic

Propositional Logic

Based on propositions and relationships between them

Propositions: p, q, r, ...

Relationships:

AND	$p \wedge q$
OR	$p \vee q$
NOT	$\neg p$
implies	$p \Rightarrow q$
...	...

Can you represent this example in Propositional logic?

- Alice is taller than Bob
- Bob is taller than Charlie
- Therefore Alice is taller than Charlie

Atomic propositions (p, q) cannot see inside the statement and make relationships between the objects

First-order Logic

Background
Propositional vs First-order Logic

Propositional Logic

Atomic propositions (p, q) cannot see inside the statement and make relationships between objects.

First-order Logic (Predicate Logic)

Based on Propositional Logic but with the addition of non-logical objects, predicates, and qualifiers

- **Objects:** x, y, Alice (Constants/Variables)
- **Predicates:** taller: $T(x)$
- **Quantifiers:**
  - $\forall$
  - $\exists$

The Solution:

- $\forall x \forall y \forall z((T(x, y) \land T(y, z)) \rightarrow T(x, z))$
- $T(a, b)$
- $T(b, c)$
- $\therefore T(a, c)$

Background
Propositional vs First-order Logic

Propositional Logic

Atomic propositions (p, q) cannot see inside the statement and make relationships between objects.

First-order Logic (Predicate Logic)

Based on Propositional Logic but with the addition of non-logical objects, predicates, and qualifiers

- **Objects:** x, y, Alice (Constants/Variables)
- **Predicates:** taller: $T(x)$
- **Quantifiers:**
  - $\forall$
  - $\exists$

The Solution:

- $\forall x \forall y \forall z((T(x, y) \land T(y, z)) \rightarrow T(x, z))$
- $T(a, b)$
- $T(b, c)$
- $\therefore T(a, c)$

Background
Propositional vs First-order Logic

Propositional Logic

Atomic propositions (p, q) cannot see inside the statement and make relationships between objects.

First-order Logic (Predicate Logic)

Based on Propositional Logic but with the addition of non-logical objects, predicates, and qualifiers

- **Objects:** x, y, Alice (Constants/Variables)
- **Predicates:** taller: $T(x)$
- **Quantifiers:**
  - $\forall$
  - $\exists$

The Solution:

- $\forall x \forall y \forall z((T(x, y) \land T(y, z)) \rightarrow T(x, z))$
- $T(a, b)$
- $T(b, c)$
- $\therefore T(a, c)$

"Hard" logic is not differentiable

1. Can we integrate declarative rules with end-to-end neural network training?

2. Can such rules help ease the need for data?

3. How does incorporating domain expertise compare against large training resources powered by pre-trained representations?

Contributions

Contributions

1. A "Compiler" for Logic: They propose a method to automatically translate first-order logic rules into a differentiable neural network sub-graph.
2. Zero-Parameter Augmentation: The logic sub-graph has no learnable weights. It is purely deterministic.
3. End-to-End Training: Because the logic is differentiable you can train the whole system using standard backpropagation.

Challenges & Method

NN x Logic Challenges

Logic is not differentiable

NNs are acyclic

Mapping predicates to neurones

NN x Logic Challenges

Logic is not differentiable

NNs are acyclic

Mapping predicates to neurones

Differentiable Logic

Logic is not differentiable

Łukasiewicz T-norm and T-conorm Logics

A type of fuzzy "soft" logic.

- True/False $\rightarrow$ Continuous values in $[0, 1]$.
- Predicates $P(x) \rightarrow$ Neurons/Vectors p.
- AND $(A \wedge B) \rightarrow \max(0, A + B - 1)$
- OR $(A \vee B) \rightarrow \min(1, A + B)$
- NOT $(\neg A) \rightarrow 1 - A$...

For example:
Let h be the neuron output for "Human"
Let m be the neuron output for "Mortal" The rule $\forall x(Human(x) \rightarrow Mortal(x))$ is converted into an "edge" in the network: $\min(1, 1 - h + m)$

Differentiable Logic

Logic is not differentiable

Constraints Beget Distance Functions

Antecedent	Distance $d(z)$
$\bigwedge_i Z_i$	$max(0, \sum_i z_i - \lvert Z \rvert + 1)$
$\bigvee_i Z_i$	$min(1, \sum_i z_i)$
$\neg \bigvee_i Z_i$	$max(0, 1 - \sum_i z_i)$
$\neg \bigwedge_i Z_i$	$min(1, \lvert Z \rvert - \sum_i z_i)$

General case example:

- $(\neg A \vee B) \wedge (C \vee D)$
- transformed into $P \wedge Q$
- $(\neg A \vee B) \leftrightarrow P$
- $(\neg C \vee D) \leftrightarrow Q$

NN x Logic Challenges

Logic is not differentiable

NNs are acyclic

Mapping predicates to neurones

NN x Logic Challenges

Logic is not differentiable

NNs are acyclic

Mapping predicates to neurones

Acyclic Logic

NNs are acyclic

Cyclicity of constrains

Definition: Let G be a computation graph. An implicative statement $L \rightarrow R$ is cyclic with respect to G if, for any literal $R_i \in R$, the node r_i associated with it is **upstream** of the node l_j associated with some literal $L_j \in L$. An implicative statement is acyclic if it is not cyclic.

Sometimes conversionas are possible: $B \rightarrow A$ is equivilent to $\neg A \rightarrow \neg B$

NN x Logic Challenges

Logic is not differentiable

NNs are acyclic

Mapping predicates to neurones

NN x Logic Challenges

Logic is not differentiable

NNs are acyclic

Mapping predicates to neurones

Predicates into Neurons

Mapping predicates to neurones

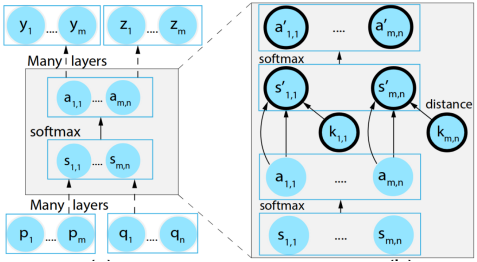
1. Identify specific neurons ("named neurons") in the network that correspond to logical concepts
2. Add a parallel network to the original NN that takes those neurons as input and computes the "Rule Satisfaction" score using the Lukasiewicz formulas
3. modify the forward pass with this nex value:
   - original neuron output: $y = \sigma(Wx)$
   - augmented output: $y = \sigma(Wx + \rho d(z))$

Method
Tasks

3 different tasks:

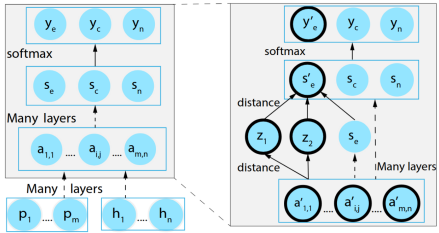
1. For Machine Comprehension: They used BiDAF
2. For NLI (Natural Language Inference): They used Decomposable Attention (DAtt)
3. For Text Chunking: They used a Bi-LSTM

Method
Tasks



BiDAF

$$R2 : \forall i, j \in C, K_{i,j} \vee A_{i,j} \rightarrow A_{i,j}$$



LDatt

$$N3 : Z_1 \wedge Z_2 \rightarrow \neg Y'_{entail}, \text{ where}$$
$$\exists j \in C, \neg(\exists i \in C, \overleftarrow{A'_{i,j}}) \leftrightarrow Z_1,$$
$$\exists j \in C, \neg(\exists i \in C, \overrightarrow{A'_{i,j}}) \leftrightarrow Z_2.$$

Experiments & Results

Results
Machine Comprehension: augmenting intermediate decisions

Experiments params:

- Score represents the average span F1 on the test set among 3 random runs
- $\rho = 2$
- $R1 : \forall i, j \in C, K_{i,j} \rightarrow A_{i,j}$
- $R2 : \forall i, j \in C, K_{i,j} \vee A_{i,j} \rightarrow A_{i,j}$

	%Train	BiDAF	$+R_1$	$+R_2$	$+ELMo$	$+ELMo, R_1$	
+4.0	10%	57.5	**61.5**	60.7	71.8	**73.0**	+1.2
+1.5	20%	65.7	**67.2**	66.6	76.9	**77.7**	+0.8
+2.0	40%	70.6	**72.6**	71.9	80.3	**80.9**	+0.6
+1.7	100%	75.7	**77.4**	77.0	83.9	**84.1**	+0.2

Results
Machine Comprehension: augmenting intermediate decisions

Experiments params:

- Score represents the average span F1 on the test set among 3 random runs
- $\rho = 2$
- $R1 : \forall i, j \in C, K_{i,j} \rightarrow A_{i,j}$
- $R2 : \forall i, j \in C, K_{i,j} \vee A_{i,j} \rightarrow A_{i,j}$

%Train	BiDAF	$+R_1$	$+R_2$	$+ELMo$	$+ELMo, R_1$
10%	57.5	**61.5**	60.7	71.8	**73.0**
20%	65.7	**67.2**	66.6	76.9	**77.7**
40%	70.6	**72.6**	71.9	80.3	**80.9**
100%	75.7	**77.4**	77.0	83.9	**84.1**

-3.1

-3

Results
Machine Comprehension: augmenting intermediate decisions

Experiments params:

- Score represents the average span F1 on the test set among 3 random runs
- $\rho = 2$
- $R1 : \forall i, j \in C, K_{i,j} \rightarrow A_{i,j}$
- $R2 : \forall i, j \in C, K_{i,j} \vee A_{i,j} \rightarrow A_{i,j}$

%Train	BiDAF	$+R_1$	$+R_2$	$+ELMo$	$+ELMo, R_1$
10%	57.5	**61.5**	60.7	71.8	**73.0**
20%	65.7	**67.2**	66.6	76.9	**77.7**
40%	70.6	**72.6**	71.9	80.3	**80.9**
100%	75.7	**77.4**	77.0	83.9	**84.1**

Results

Natural Language Inference: augmenting output decisions constrained by intermediate states

Experiments params:

- Score represents the average accuracy on the test set among 3 random runs
- For $N1$ and $N2$ $\rho = (8, 8, 8, 8, 4)$ and for $N3$ $\rho = (2, 2, 1, 1, 1)$
- $N1 : \forall i, j \in C, K_{i,j} \rightarrow A_{i,j}$
- $N2 : \forall i, j \in C, K_{i,j} \vee A_{i,j} \rightarrow A_{i,j}$
- $N3 : Z_1 \wedge Z_2 \rightarrow \neg Y'_{entail}$, where

  $\exists j \in C, \neg(\exists i \in C, \overleftarrow{A}'_{i,j}) \leftrightarrow Z_1,$

  $\exists j \in C, \neg(\exists i \in C, \overrightarrow{A}'_{i,j}) \leftrightarrow Z_2.$

% Train	L-DAtt	+N1	+N2	+N3	+N2,3
1%	61.2	**64.9**	63.9	62.5	64.3
2%	66.5	**70.5**	69.8	67.9	70.2
5%	73.4	76.2	**76.6**	74.0	76.4
10%	78.9	80.1	**80.4**	79.3	80.3
100%	**87.1**	86.9	**87.1**	87.0	86.9

Results

Text Chunking: augmenting output decisions constrained using label dependencies

Experiments params:

- Score represents the average accuracy on the test set among 3 random runs
- For $C1:4$ $\rho = 4$ and for $N5$ $\rho = 16$

% Train	BiLSTM	+CRF	+C1:5	+CRF,C1:5
5%	87.2	86.6	**88.9**	88.6
10%	89.1	88.8	**90.7**	90.6
20%	90.9	90.8	**92.1**	**92.1**
40%	92.5	92.5	93.4	**93.5**
100%	94.1	94.4	94.8	**95.0**

What I liked

- Diversity of experiments
- Logic inside the architecture
- Good generalizability

What could be changed

- It's not usually easy to map logic to a specific neuron
- Simple logic formulas