Report for CSE 505 - Phase 3

Xiaofei Sun
Stony Brook University
ID:111753600
Email: http://www.michaelshell.org/contact.html

***Abstract*—My abstract**

Keywords—Multi-valued Real Logic, Tensor Network, Reasoning.

I. Paper Overview

A. Basic Information

Here is some basic information about the paper I selected:

- Title: Logic Tensor Networks: Deep Learning and Logical Reasoning from Data and Knowledge[1].
- Author: Luciano Serafini and Artur dAvila Garcez
- From: ArXiv.2016
- Abstract: This paper proposes Logic Tensor Networks (LTN) to integrate **learning** and **reasoning** together based on vector representation. The proposed model LTN represent each object with a vector and then converts each function on multiple objects into a manipulate on their vector representations. Then it also uses a s-norm operator to transform a predicate to a real number in $[0, 1]$, which means the confidence of this predicate. Finally, it defines a lose function for all predicates, which means it transfers the reasoning process into a learning and optimization problem.

B. Definitions

Recall that a first-order language L is composed by three parts:

- $\mathcal{C} = \{c_1, c_2, \dots\}$, the set of constant symbols;
- $\mathcal{F} = \{f_1, f_2, \dots\}$, the set of functional symbols;
- $\mathcal{P} = \{p_1, p_2, \dots\}$ the setof predicate symbols.

C. Groundings

LTN defines a term $\mathcal{G}$, called **grounding**, which contains $\mathcal{G}(c)$, $\mathcal{G}(f)$, $\mathcal{G}(P)$ respect to $c \in \mathcal{C}$, $f \in \mathcal{F}$, $P \in \mathcal{P}$:

Also, LTN define the manipulate of each clause ϕ.

1) Constant: For each constant, LTN allocate a n-dimension vector to it:

$$\mathcal{G}(c) \in \mathbb{R}^n \tag{1}$$

2) Function: For each function f/m on m parameters, LTN define it as a mapping on vector space, which means:

$$\mathcal{G}(f(t_1, t_2, \dots, t_m)) \in \mathbb{R}^{mn} \to \mathbb{R}^n \tag{2}$$

Note that here we specify the parameter of f is t_i instead of c_i because function f can be recursive like $f(f_1(c1, c2), f_2(c1, c3)$.

$$\mathcal{G}(f(t_1, t_2, \dots, t_m)) = \mathcal{G}(f)\mathcal{G}(t_1), \mathcal{G}(t_2), \dots, \mathcal{G}(t_m) \tag{3}$$

More specifically, $\mathcal{G}(f(t_1, t_2, \dots, t_m))$ is fitted by a linear function, which can be implemented with tensor network:

$$\begin{aligned}\mathcal{G}(f(t_1, t_2, \dots, t_m)) =& \mathcal{G}(f)(v_1, v_2, \dots, v_m)\\ =& \mathcal{G}(f)(v)\\ =& M_f v + N_f\end{aligned} \tag{4}$$

where $v = \langle v_1, v_2, \dots, v_m \rangle$, $M_f \in \mathbb{R}^{n \times mn}$, and $N_f \in \mathbb{R}^n$

3) Predicate: Like the grounding of a function,

$$\mathcal{G}(P(t_1, t_2, \dots, t_m)) \in \mathbb{R}^{mn} \to [0, 1] \tag{5}$$

Again, as t_i has been mapped to a vector, then:

$$\mathcal{G}(P(t_1, t_2, \dots, t_m)) = \mathcal{G}(P)\mathcal{G}(t_1), \mathcal{G}(t_2), \dots, \mathcal{G}(t_m) \tag{6}$$

Therefore we transfer a predicate into a series of manipulation on tensor space:

$$\begin{aligned}\mathcal{G}(P(t_1, t_2, \dots, t_m)) =& \mathcal{G}(f)(v_1, v_2, \dots, v_m)\\ =& \mathcal{G}(f)(v)\\ =& \sigma(u_p^T tanh(v^T W_P v + V_P v + B_P))\end{aligned} \tag{7}$$

where $v = \langle v_1, v_2, \dots, v_m \rangle \in \mathbb{R}^{mn}$, $W_P \in \mathbb{R}^{mn \times mn \times k}$, $V_P \in \mathbb{R}^{mn \times k}$, $B_P \in \mathbb{R}^k$, $u_P \in \mathbb{R}^k$, and σ is the sigmoid function.

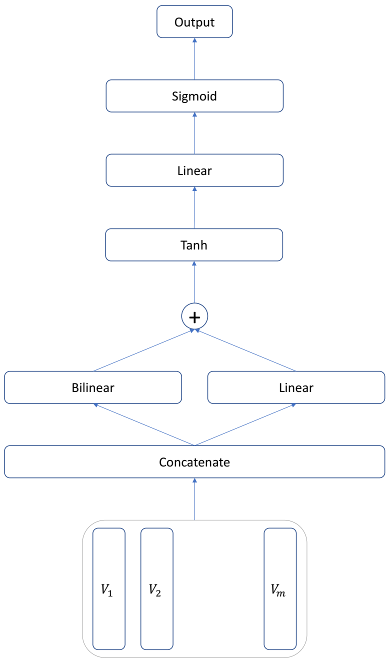


Figure 1: Model for Predicate

4) Clause: In this project, we assume each clause is disjunctive normal form. So it's easy to get that

$$\mathcal{G}(\phi_1, \phi_2, \ldots, \phi_k) = \mu(\mathcal{G}(\phi_1), \mathcal{G}(\phi_2), \ldots, \mathcal{G}(\phi_k)) \tag{8}$$

where μ is the max function.

D. Optimization

So now we get the definition of all components of multi-valued first-order language in tensor networks. The next step is to define a proper lose function.

Saying we know the value of $\phi(x)$ in our dataset, where x is a constant. Intuitively, an optimal grouding should make $\mathcal{G}(\phi(t))$ as close as to $\phi(x)$.

II. Models

A. Logic Tensor Network (LTN)

B. Weight Logic Tensor Network (WLTN)

C. Convolutional Logic Tensor Network (CLTN)

III. Experiments

A. Fit Knowledge Base

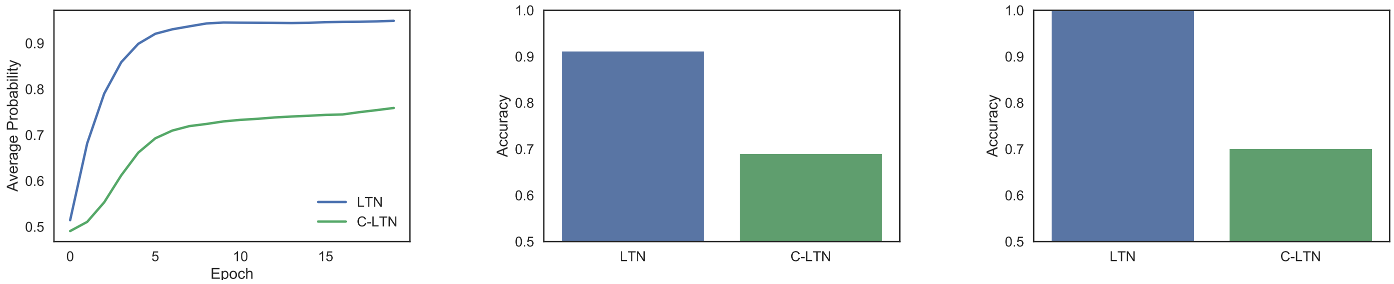
B. Learn From Rule

C. Parameter Sensitive

IV. Details and Discussion

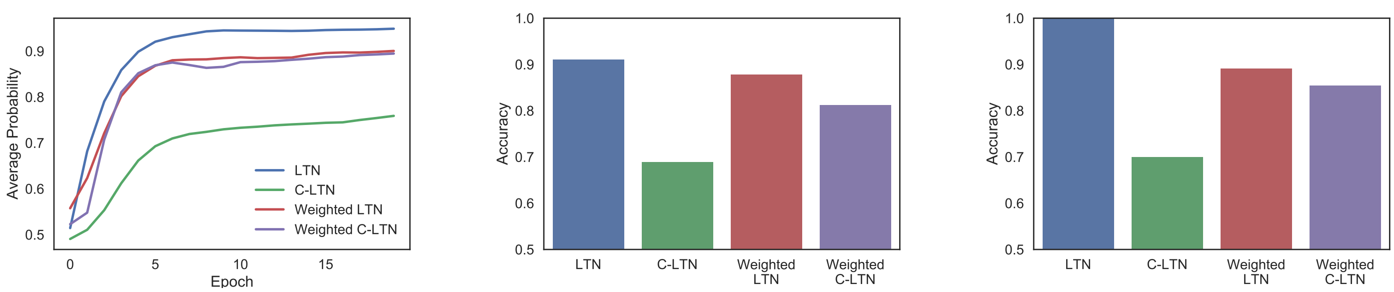
References

[1] L. Serafini and A. d. Garcez, "Logic tensor networks: Deep learning and logical reasoning from data and knowledge," *arXiv preprint arXiv:1606.04422*, 2016.



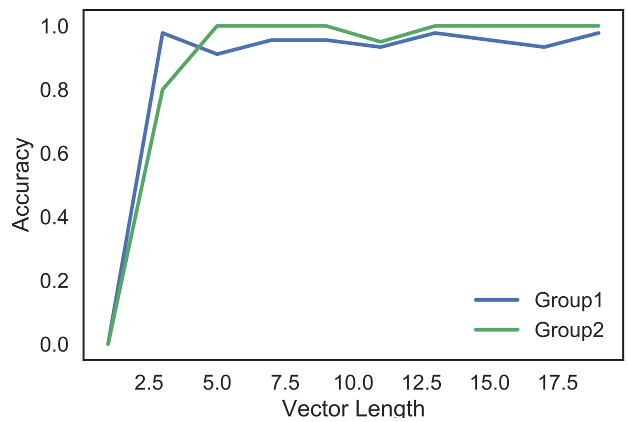
(a) Probability w.r.t. Epoch (b) Best Accuracy on Group1 (c) Best Accuracy on Group2

Figure 2: Fitting Observed Facts.

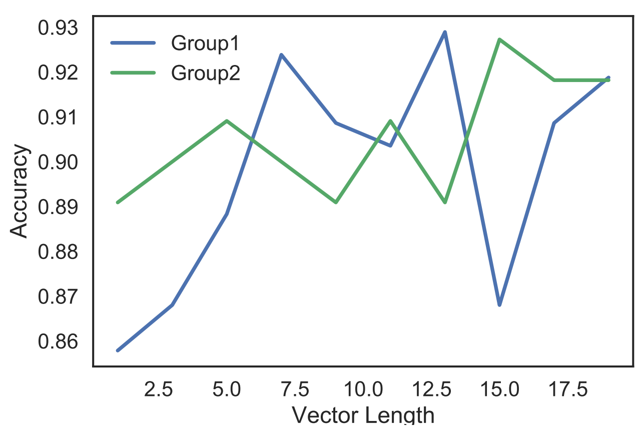


(a) Probability w.r.t. Epoch (b) Best Accuracy on Group1 (c) Best Accuracy on Group2

Figure 3: Learning from Observed Facts & Rules.



(a) Fitting Observed Facts



(b) Learning from Rules

Figure 4: Best Accuracy w.r.t. Vector Length.