Reasoning with propositional knowledge based on fuzzy neural logic

Wangming Wu
Hoon-Heng Teh
Bo Yuan
Reasoning with Propositional Knowledge
Based on Fuzzy Neural Logic

Wangming Wu and Hoon-Heng Teh
Institute of Systems Science, National University of Singapore,
Singapore 0511

Bo Yuan
Department of Systems Science and Industrial Engineering, Thomas J.
Watson School of Engineering, Binghamton University (SUNY),
Binghamton, New York 13902

In this article, a new kind of reasoning for propositional knowledge, which is based on
the fuzzy neural logic initiated by Teh, is introduced. A fundamental theorem is pre-
"sented showing that any fuzzy neural logic network can be represented by operations:
bounded sum, complement, and scalar product. Propositional calculus of fuzzy neural
logic is also investigated. Linear programming problems risen from the propositional
calculus of fuzzy neural logic show a great advantage in applying fuzzy neural logic to
answer imprecise questions in knowledge-based systems. An example is reconsidered
here to illustrate the theory.

I. INTRODUCTION

Fuzzy logic, attributable to Zadeh,1 is basically a logic for approximate
reasoning. As a generalization of classical two-valued logic, fuzzy logic extends
the range of truth value from {0, 1} to [0, 1]. The basic logical connectives
AND, OR, and NOT in fuzzy logic are interpreted as t-norms, t-conorms, and
complements (e.g., Wu2). By such a generalization, a classical two-valued logic
system becomes a special case of fuzzy logic systems.

In recent years, a new kind of logic named fuzzy neural logic has been
introduced by Teh.3 Informally, a fuzzy neural logic is a kind of continuous-
valued logic that includes the bounded weighted averages as its basic operation.
Notice that a bounded weighted average (or convex combination) is an aggre-
gate operation used in many practical problems. However, a bounded weighted
average is not used to interpret AND or OR logical connectives, because they
are not associative.

This article shows that every operation in Kleene–Lukasiewicz fuzzy logic
can be implemented by some simple fuzzy neural logic networks. This means
Kleene–Lukasiewicz fuzzy logic is a special case of fuzzy neural logic dis-
cussed here. Fuzzy neural logic has a bounded weighted average as its addi-
tional operation that is not included in Kleene–Lukasiewicz fuzzy logic. In this article, we present a fundamental theorem showing that any basic neuron in fuzzy neural logic network can be represented by compositions of Kleene–Lukasiewicz logical operations and scalar product operations. Therefore, fuzzy neural logic is essentially a weighted Kleene–Lukasiewicz logic.

In the literature, some work has been done on applying Boolean programming$^4$ to answer precise or imprecise questions in the framework of propositional knowledge. However, statements in those work are formulated as equations or inequalities involving Boolean variables. In this article, by using fuzzy neural logic, the linear programming approach, instead of the Boolean programming approach, is employed to represent and reason with propositional knowledge involving fuzzy variables.

In the next section, a brief sketch of fuzzy neural logic network and two important theorems, the fundamental theorem and the dual theorem of fuzzy neurons, are presented. Section III establishes the propositional calculus for fuzzy neural logic and shows how to formulate problems in fuzzy propositional logic into linear programming problems. In Section IV, an example given in Castro et al.$^5$ is reconsidered to illustrate the new method. Finally, some conclusions are discussed.

II. FUZZY NEURAL LOGIC

Fuzzy neural logic is a kind of fuzzy logic whose logical operations can be implemented by neural networks. Each node in the network represents a premise (input nodes) and/or a conclusion (output nodes). A simple logical rule is represented by a neuron with a set of input nodes and an output node in Figure 1.

In Figure 1, there are $n$ input nodes with the values of activation $a_1$, $a_2$, . . . , $a_n$ in [0, 1], respectively. An output node is connected with the input node corresponding to $a_i$ with weight $w_i$, where $w_i$ are real numbers. The output value $b$ depends on values of $a_1$, $a_2$, . . . , $a_n$, $w_1$, . . . , $w_n$ and a slope

(a) A neuron for simple logical rule

(b) A slope threshold function $f$

Figure 1. A neuron for simple logical rule and slope threshold function $f$
threshold function $f$, which is defined by

$$f(x) = \begin{cases} 
1 & \text{if } x > 1 \\
 x & \text{if } 0 \leq x \leq 1 \\
0 & \text{if } x < 0 
\end{cases}$$

This function is also called the *linear threshold function*. Clearly, $b \in [0, 1]$. In particular, we can implement the bounded sum $\oplus$, the bounded product $\otimes$, the Lukasiewicz implication $\Rightarrow$, and the complement $c^*$ by following neurons shown in Figure 2.

Observe that all connection weights in the neurons of Figure 2 are either 1 or $-1$. Let $\vee = \max$ and $\wedge = \min$. It is known that $a_1 \vee a_2 = a_1 \oplus (a_1^c \otimes a_2)$ and $a_1 \wedge a_2 = a_1 \otimes (a_1^c \oplus a_2)$. Therefore, Kleene’s logical operations $\vee$ and $\wedge$ can also be implemented by neurons with connection weights either 1 or $-1$.

In Figure 1, if $n = 1$ and $w_i \in [0, 1]$, then $a_1w_1 = f(a_1w_1) \in [0, 1]$. That means whenever $a_1, w_1 \in [0, 1]$, the scalar product $a_1w_1$ can be implemented by another neuron which has an input node with the value of activation $a_1$ and an output node connected with the input node with weight $w_1$. (Fig. 3)

*Bounded sum $\oplus$, bounded product $\otimes$, Lukasiewicz implication $\Rightarrow$, and standard complement $c$ are defined, respectively, by the following: $a \oplus b = \min(a + b, 1); a \otimes b = \max(a + b - 1, 0); a \Rightarrow b = \min(1, 1 - a + b)$, and $a^c = 1 - a$, for any $a, b \in [0, 1]$.***
In general, if $w_1, \ldots, w_n$ are real numbers, then there exists a natural number $N = 1 + \max(|w_1|, \ldots, |w_n|)$ such that $w_i = c_i N$, where $[x]$ means the integer part of real number $x$ and $c_i = w_i / N \in [-1, 1]$, $i = 1, \ldots, n$. Then we have $f(\sum_{i=1}^{n} a_i w_i) = f(N \sum_{i=1}^{n} a_i c_i) = f(N \sum_{i=1}^{n} a_i |c_i| sgn(c_i)) \in [0, 1]$. Therefore, any neuron can be decomposed as a combination of scalar product operations and the simple neurons with connection weights either 1 or $-1$. (Fig. 4)

Note that the slope threshold function $f$ can be represented by $f(x) = \max(\min(x, 1), 0) = \min(\max(x, 0), 1)$. Thus, we have

$$f(\sum_{i=1}^{n} a_i w_i) = \max(\min(\sum_{i=1}^{n} a_i w_i, 1), 0) = \min(\max(\sum_{i=1}^{n} a_i w_i, 0), 1)$$

Therefore, $f(a_1 - a_2) = \max(\min(a_1 - a_2, 1), 0) = \max(a_1 - a_2, 0) = a_1 \otimes a_2^c$ and $f(a_1 + a_2) = a_1 \oplus a_2$. Generally, we have the following lemma.

**Lemma 1.** For any real number $x$ and any $a \in [0, 1]$,

$$f(x + a) = (f(x) \oplus a) \otimes (f(-x))^c \quad (2)$$

and

$$f(x - a) = (f(x) \otimes a^c) \oplus f(x - 1). \quad (3)$$

**Proof.** We only prove Eq. 2 here. Equation 3 can be proved in a similar way. By Eq. 1, we have

$$f(x + a) = \begin{cases} 
1 & \text{if } x + a > 1 \\
 x + a & \text{if } 0 \leq x + a \leq 1 \\
0 & \text{if } x + a < 0
\end{cases}$$

when $x + a > 1$, $x > 1 - a$ and $-x < a - 1 < 0$. Then $f(x) \geq f(1 - a) = 1 - a$ and $f(-x) = 0$. Thus, $(f(x) \oplus a) \otimes (f(-x))^c = f(x) \oplus a = \min(f(x) + a, 1) = 1 = f(x + a)$. When $x + a < 0$, $x < -a < 0$ and $-x > a$, then $f(x) = 0$ and $f(-x) \geq f(a) = a$. $(f(x) \otimes a) \otimes (f(-x))^c \leq a \otimes a^c = 0$. Thus, $(f(x) \oplus a) \otimes (f(-x))^c = 0 = f(x + a)$. Finally, when $0 \leq x + a \leq 1$, $-a \leq x \leq 1 - a$. If $0 \leq x \leq 1 - a$, then $f(x) = x$ and $f(-x) = 0$. Thus, $(f(x) \oplus a) \otimes (f(-x))^c = x + a = f(x + a)$. If $-a \leq x < 0$, then $f(x) = 0$ and $f(-x) = -x$. Thus, $(f(x) \oplus a) \otimes (f(-x))^c = a \otimes (x + a) = \min(a + 1 - (-x) - 1, 1) = a + x = f(a + x)$. This completes the proof.

Now for any $a \in [0, 1]$, $f(a) = a = a \oplus (a \otimes a^c)$, $f(-a) = 0 = a \otimes a^c$. Thus, $f(aw)$ can be represented by compositions of logic operation $\oplus$, $\otimes$ and $^c$, when $w$ is either 1 or $-1$. Assume that $w_1, \ldots, w_n \in \{-1, 1\}$ and $f(\sum_{i=1}^{n} a_i w_i)$ can
Figure 4. An arbitrary neuron represented by scalar products and neurons with connection weights either 1 or -1. 

be represented by compositions of logical operations $\Theta$, $\otimes$ and $\circ$ Then, by Lemma 1,

$$f\left(\sum_{i=1}^{n} a_i w_i + a_{n+1}\right) = \left(f\left(\sum_{i=1}^{n} a_i w_i\right) \oplus a_{n+1}\right) \otimes f\left(\sum_{i=1}^{n} a_i (-w_i)\right),$$

and

$$f\left(\sum_{i=1}^{n} a_i w_i - a_{n+1}\right) = f\left(\sum_{i=1}^{n} a_i w_i\right) \otimes a_{n+1}^c \oplus f\left(\sum_{i=1}^{n} a_i w_i - 1\right).$$

If $w_1 = \cdots = w_n = -1$, then

$$f\left(\sum_{i=1}^{n} a_i w_i - a_{n+1}\right) = f\left(\sum_{i=1}^{n} a_i w_i\right) \otimes a_{n+1}^c.$$

Otherwise, there exists one weight, say $w_1 = 1$. Then we have

$$f\left(\sum_{i=1}^{n} a_i w_i - a_{n+1}\right) = \left(f\left(\sum_{i=1}^{n} a_i w_i\right) \otimes a_{n+1}^c\right) \oplus f\left(a_1(-1) + \sum_{i=2}^{n} a_i w_i\right).$$

Therefore,

$$f\left(\sum_{i=1}^{n} a_i w_i + a_{n+1}\right)$$

and

$$f\left(\sum_{i=1}^{n} a_i w_i - a_{n+1}\right)$$
The dual of a neuron can also be represented by compositions of logical operations $\oplus$, $\otimes$, and $\cdot$. Since $a_1 \otimes a_2 = (a_1^\oplus a_2)^\cdot$, we have the following fundamental theorem.

**Theorem 1.** (the fundamental theorem). Every fuzzy neural logic network can be represented by the composition of logical operations $\oplus$, $\cdot$, and scalar product operations.

Let us consider the dual of a neuron shown in Figure 5.

In Figure 5, $d = b^c = 1 - b = 1 - \max(min(\sum_{i=1}^n a_i w_i, 1), 0) = \min(1 - \min(\sum_{i=1}^n a_i w_i, 1), 1) = \min(\max(1 - \sum_{i=1}^n a_i w_i, 0), 1) = \min(\max(\sum_{i=1}^n(1 - a_i)w_i + (1 - \sum_{i=1}^n w_i), 0), 1)$. Thus the dual of a neuron in Figure 5 can be implemented by another neural network shown in Figure 6.
III. PROPOSITIONAL CALCULUS OF FUZZY NEURAL LOGIC SYSTEMS

We are now ready to develop a propositional system of fuzzy neural logic. The symbols, with or without subscripts or primes, such as $X, Y, X', X^n$ that are used to denote propositions (i.e., sentences) are called atomic formulas, or atoms for short. From propositions, we can build compound propositions by using logical connectives. In fuzzy neural logic, we use five logical connectives: \( \neg \) (negation), \( \& \) (conjunction), \( \triangledown \) (disjunction), \( \rightarrow \) (implication), \( \leftrightarrow \) (equivalence), and a class of scalar product connective $p_\alpha$, where $\alpha \in [0, 1]$. These connectives can be used to build compound propositions from atoms. More generally, we can construct more complicated compound propositions by applying them repeatedly. These compound propositions are called well-formed formulas. More formally, we give the following definition.
**Definition 1.** Well-formed formulas, or formulas for short, in the propositional system of fuzzy neural logic, are defined recursively as follows:

1. An atom is a formula.
2. If $A$ is a formula, then $(\neg A)$ and $(p_{\alpha}A)$ are formulas ($\alpha \in [0, 1]$).
3. If $A$ and $B$ are formulas, then $(A \& B)$, $(A \not\equiv B)$, $(A \rightarrow B)$ and $(A \leftrightarrow B)$ are formulas.
4. All formulas are generated by using the above rules.

**Remark 1.** As usual, some pairs of parentheses in a formula may be dropped when no confusion is possible, for example, $A \rightarrow \neg B$ means $(A \rightarrow (\neg B))$.

**Remark 2.** The set of all formulas is denoted as $\Lambda$.

**Definition 2.** A mapping $\nu$ from $\Lambda$ to $[0, 1]$ is said to be a valuation if for any formulas $A$, $B$ and any $\alpha \in [0, 1]$,

1. $\nu(\neg A) = (\nu(A))^c$;
2. $\nu(p_{\alpha}A) = \alpha \nu(A)$;
3. $\nu(A \& B) = \nu(A) \odot \nu(B)$;
4. $\nu(A \not\equiv B) = \nu(A) \oplus \nu(B)$;
5. $\nu(A \rightarrow B) = \nu(A) \Rightarrow \nu(B)$;
6. $\nu(A \leftrightarrow B) = [\nu(A) \Rightarrow \nu(B)] \odot [\nu(B) \Rightarrow \nu(A)]$.

$(\Lambda, \nu)$ is called a propositional system of fuzzy neural logic.

Note that a valuation of a formula $A$ is determined by the assignment of truth values for atoms occurring in $A$. For instance, let $A = p_{0.3}X \rightarrow ((X \rightarrow Y) \not\equiv Y)$, where $X$ and $Y$ are atoms. By definition, we have $\nu(A) = 0.3 \nu(X) \Rightarrow [(\nu(X) \Rightarrow \nu(Y)) \oplus \nu(Y)^c]$. If we take $\nu(X) = 0.7$ and $\nu(Y) = 0.5$, then $\nu(A) = 0.21 \Rightarrow [(0.7 \Rightarrow 0.5) \oplus 0.5] = 0.21 \Rightarrow (0.8 \oplus 0.5) = 1$. In general, for an arbitrary valuation $\nu$, we have $\nu(A) = (0.3\nu(X))^c \oplus (\nu(X))^c \oplus \nu(Y) \oplus \nu(Y)^c = (0.3\nu(X))^c \oplus 1 = 1$. Therefore, formula $A$ will be a tautology. We now give a formal definition of $\lambda$-tautologies.

**Definition 3.** Let $\lambda \in [0, 1]$, A formula $A$ is called a $\lambda$-tautology and denoted as $\vdash_\lambda A$ if for any valuation $\nu$, $\nu(A) \geq \lambda$. A tautology $A$ is a 1-tautology and simple denote as $\vdash A$.

**Theorem 3.** In $(\Lambda, \nu)$, for any formula $A$, $B$, $C$ in $\Lambda$ and for any $\alpha$, $\beta \in [0, 1]$, the followings hold.

1. $\vdash A \not\equiv \neg A$.
2. $\vdash (A \rightarrow B) \leftrightarrow (\neg A \rightarrow \neg B)$.
3. $\vdash (A \& (A \rightarrow B)) \rightarrow B$.
4. $\vdash A \& B$ if and only if $\vdash A$ and $\vdash B$.
5. $\vdash p_{\alpha} A \rightarrow A$.
6. $\vdash p_{\alpha}(p_{\beta}A) \leftrightarrow p_{\alpha\beta}A$.
7. $\vdash p_{(\alpha\beta)+\gamma} A \rightarrow (p_{\alpha}A \not\equiv p_{\beta}A)$. 
\[(8) \equiv (p_{aA} \land p_{\beta A}) \rightarrow (p_{a\wedge \beta} A \land A).\]
\[(9) \equiv_{A} A \equiv (\neg A \land \neg A).\]
\[(10) \text{If } a \neq 0, \text{ then } \equiv_{a} p_{aA} \text{ if and only if } \equiv_{a} A.\]

**Proof.** (1)–(4) are true in Lukasiewicz logic. To prove (5), let \(\nu\) be an arbitrary valuation and \(a = \nu(A)\) then \(\nu(p_{aA} \rightarrow A) = \alpha a \Rightarrow a = 1.\) (6) holds because for any valuation, \(\nu(p_{a}(p_{\beta A})) = \alpha \beta a = \nu(p_{a\beta} A).\) To prove (7), for any valuation \(\nu,\)
\(\nu(p_{aA} \equiv p_{\beta A}) = \alpha a \oplus \beta a = \min((\alpha + \beta)a, a) = (\alpha \oplus \beta)a = \nu(p_{a\oplus \beta} A).\) Similarly, \(\nu(p_{a\wedge \beta} A \land A) = \max(\alpha \wedge \beta a + a - 1, 0) = \max(\max((\alpha + \beta - 1)a, 0) + a - 1, 0) = \max((\alpha + \beta - 1)a + a - 1, 0) = \max((\alpha + \beta)a - 1, 0) = \nu(p_{aA} \land p_{\beta A}).\) Thus, (8) is obtained. (9) follows from the fact that \(\nu(A \equiv (\neg A \land \neg A)) = \nu(A) \oplus (\nu(A) \land \nu(A)) = \nu(A) \land (\nu(A) \lor \nu(A))^c \geq 0.5,\) for any valuation \(\nu.\) Finally, for any valuation \(\nu, \nu(p_{aA}) \equiv \alpha \iff \nu(A) \equiv \alpha \iff \nu(A) = 1.\) (10) holds.

**Definition 4.** Suppose \(A\) and \(B\) are formulas, \(A\) and \(B\) are said to be equivalent and denoted by \(A \equiv B\) if \(\equiv A \leftrightarrow B.\)

**Remark 3.** It is easy to see that \(A \equiv B\) if and only if for any valuation \(\nu, \nu(A) = \nu(B).\) Besides, in Theorem 3, (2) and (6) can be rewritten as \(A \rightarrow B = B \rightarrow \neg A,\)
\(p_{a} p_{\beta A} = p_{a \beta} A,\) respectively.

**Definition 5.** Let \(X\) be an atom. Denote \(\perp\) as \(p_{0} X\) and, \(T\) as \(\neg \perp.\) For \(n \geq 1,\) define a \(n\)-ary connective \(M_{n}\) as \(M_{n}(A_{1}, \ldots, A_{n}) = p_{1/n} A_{1} \equiv \cdots \equiv p_{1/n} A_{n}.\)

**Remark 4.** For any tautology \(A,\) we have \(T = A\) and \(\perp = \neg A.\) For any formula \(B, \perp \) as \(p_{0} B.\) Denote \(p_{a} T = T_{a}\) and \(\neg p_{a} A.\)\) Furthermore, \(\neg(T_{a}) = T_{1-a}.\) Obviously, \(T_{1} = T, T_{0} = \perp\) and, \(\nu(M_{n}(A_{1}, \ldots, A_{n})) = (\nu(A_{1}) + \cdots + \nu(A_{n}))/n.\)

**Theorem 4.** In \((\Lambda, \nu),\) the followings hold: for any \(A\) in \(\Lambda\) and \(\alpha, \beta\) in \([0, 1],\)

\[(11) \equiv_{a} A = \neg A \equiv \perp_{1-a} A;\]
\[(12) p_{a}(-A) = T_{a} \land \neg p_{a} A;\]
\[(13) p_{a}(-p_{\beta A}) = T_{a} \land \neg p_{a \beta} A;\]
\[(14) \equiv_{a} A \text{ if and only if } \equiv_{a} T_{a} \rightarrow A.\]

**Proof.** For any valuation \(\nu,\) let \(a = \nu(A),\) we have \(\nu(-A \equiv \perp_{1-a} A) = \min((1 - a) + (1 - a) \alpha a, 1) = 1 - a \alpha a = \nu(-p_{a} A).\) (11) holds. Further, \(\nu(p_{a}(-A)) = \alpha(1 - a) = \alpha - a \alpha = \max(\alpha + 1 - a \alpha - 1, 0) = \nu(T_{a} \land -p_{a} A).\) Thus we have (12). (13) follows from the fact that \(\nu(p_{a}(-p_{\beta} A)) = \alpha - \alpha \beta a = \max(\alpha + 1 - a \beta a - 1, 0) = \nu(T_{a} \land -p_{a \beta} A).\) Finally, \(\equiv_{a} A\) is equivalent to that for any valuation, \(\nu, \alpha \Rightarrow \nu(A) = 1.\) Thus, (14) holds.

**Remark 5.** It follows from (12) of Theorem 4 that \(-p_{a} -A = T_{1-a} \equiv p_{a} A.\)

**Definition 6.** Let \(\lambda \in [0, 1]\) and \(A_{1}, \ldots, A_{n}, A\) be formulas. \(A\) is said to be a \(\lambda\)-logical consequence of \(A_{1}, \ldots, A_{n}\) (or \(A_{1}, \ldots, A_{n}\) \(\lambda\)-imply \(A\)), denoted by
(A_1, \ldots, A_n) \models \lambda \iff \text{for every valuation } \nu \text{ such that } \nu(A_1) = \cdots = \nu(A_n) = 1, \\
the inequality } \nu(A) \geq \lambda \text{ holds. Whenever } \lambda = 1, \text{ we denote } (A_1, \ldots, A_n) \models A \\
instead of (A_1, \ldots, A_n) \models_1 A.

**Theorem 5.** If } \models \lambda (A_1 & \cdots & A_n) \rightarrow A, \text{ then } (A_1, \ldots, A_n) \models \lambda A.

**Proof.** If } \models \lambda (A_1 & \cdots & A_n) \rightarrow A, \text{ then for every valuation } \nu, \nu(A_1 & \cdots & A_n) \\
\rightarrow A) = (\nu(A_1 & \cdots & A_n))^c \oplus \nu(A) = (\nu(A_1)^c \oplus \cdots \oplus \nu(A_n)^c \oplus \nu(A)) \geq \lambda. \text{ Now, } \nu \text{ is a valuation such that } \nu(A_1) = \cdots = \nu(A_n) = 1. \text{ Thus, } \\
\nu((A_1 & \cdots & A_n) \rightarrow A) = 0 \oplus \nu(A) = \nu(A) \geq \lambda. \text{ This completes the proof.}

**Remark 6.** The inverse of Theorem 5 is not true. For example, let } A_1 = X, A = X & X, \text{ where } X \text{ is an atom. If } \nu \text{ is a valuation such that } \nu(A_1) = \nu(X) = 1, \text{ then } \\
\nu(A) = \nu(X & X) = \nu(X) \otimes \nu(X) = 1. \text{ Thus, } (A_1) = A. \text{ But } A_1 \rightarrow A \text{ is not a} \\
tautology (\text{e.g., put } \nu(X) = 0.5, \text{ then } \nu(A_1 \rightarrow A) = 0.5 \rightarrow (0.5 \otimes 0.5) = 0.5 \neq 1).

**Theorem 6.** Let } \lambda \in [0, 1] \text{ and } A, B, C, A_1, \ldots, A_n, B_1, \ldots, B_n \text{ be formulas, then

\begin{align*}
(15) \text{ (Modus ponens)} & (A, A \rightarrow B) \models B. \\
(16) \text{ (Modus tollens)} & (A \rightarrow B, \neg B) \models \neg A. \\
(17) \text{ (Syllogism)} & (A \rightarrow B, B \rightarrow C) \models A \rightarrow C. \\
(18) \text{ (Mean preserving)} & (A_1 \rightarrow B_1, \ldots, A_n \rightarrow B_n) \models M_n(A_1, \ldots, A_n) \rightarrow M_n(B_1, \ldots, B_n).
\end{align*}

**Proof.** By the Theorem 5, (15) follows from (3). (16) follows from (2). Moreover, for any valuation } \nu,

\begin{align*}
\nu((A \rightarrow B) & \& (B \rightarrow C)) \rightarrow (A \rightarrow C)) \\
= & \{((\nu(A))^c \oplus \nu(B)) \otimes ((\nu(B))^c \oplus \nu(C))\}^c \oplus (\nu(A))^c \oplus \nu(C) \\
= & (\nu(A) \otimes (\nu(B))^c) \oplus (\nu(B) \otimes (\nu(C))^c) \oplus ((\nu(A))^c \oplus \nu(C)) \\
= & ((\nu(A))^c \lor (\nu(B))^c) \oplus (\nu(B) \lor \nu(C)) \\
= & (\nu(A) \land \nu(B))^c \oplus (\nu(B) \lor \nu(C)) \\
\geq & (\nu(B))^c \oplus \nu(B) = 1.
\end{align*}

Thus, we have (17).

Finally, for any valuation } \nu, \text{ let } a_i = \nu(A_i), b_i = \nu(B_i), i = 1, \ldots, n.

\begin{align*}
\nu((A_1 \rightarrow B_1) & \& \cdots \& (A_n \rightarrow B_n)) \rightarrow (M_n(A_1, \ldots, A_n) \rightarrow M_n(B_1, \ldots, B_n))) \\
= & ((a_1^c \oplus b_1) \otimes \cdots \otimes (a_n^c \oplus b_n))^c \oplus (a_1 + \cdots + a_n)/n \otimes (b_1 + \cdots + b_n)/n \\
= & (a_1 \otimes b_1) \oplus \cdots \oplus (a_n \otimes b_n) \oplus (a_1^c/n \oplus \cdots \oplus a_n^c/n) \oplus (b_1/n \oplus \cdots \oplus b_n/n) \\
= & (a_1 \otimes b_1^c/n \oplus a_1^c/n \oplus b_1/n \oplus \cdots \oplus (a_n \otimes b_n^c) \oplus a_n^c/n \oplus b_n/n \\
\geq & 1/n + \cdots + 1/n = 1,
\end{align*}
where \((a_k \otimes b_k) \oplus a_i/n \oplus b_i/n = \min[\max(a_k - b_k, 0) + (1 - (a_k - b_k))/n, 1] \geq 1/n\), for \(k = 1, \ldots, n\). Thus (12) holds.

**Definition 7.** A fuzzy literal expression is a formula with the form \(p_\alpha X\), \(\neg p_\alpha X\) or \(-p_\alpha X\), where \(X\) is an atom and \(\alpha \in [0, 1]\). A fuzzy clause is a conjunction of one or more fuzzy literal expressions. A fuzzy clause is a disjunction of one or more literal expressions. If \(P\) is a fuzzy phrase and \(C\) is a fuzzy clause, then the formula \(P \rightarrow C\) is said a simple formula.

**Remark 7.** \(X = p_1X\) and \(\neg X = \neg p_1X\) are fuzzy literal expressions. \(p_0X = \bot\) is a fuzzy clause called null clause. If \(L\) is a fuzzy literal expression, then \(\neg L\) is also a fuzzy literal expression, and vice versa. \(C\) is a fuzzy clause if and only if \(\neg C\) is a fuzzy phrase. Therefore, a simple formula \(P \rightarrow C\) is equivalent to a fuzzy clause \(\neg P \nRightarrow C\).

For example, let \(X_1, X_2, X_3\) be atoms. Then \(p_{0.3}X_1, -p_1X_2, -p_{0.6}(-X_2), p_{-1}(-X_3), \ldots\) are fuzzy literal expressions, \(p_{0.3}X_1 \& -p_{0.6}(-X_2) \& p_1(-X_3)\) and \(-p_1X_2 \& -p_{0.4}(-X_3)\) are fuzzy phrases. \((p_{0.3}X_1 \nRightarrow p_{0.6}(-X_2) \nRightarrow p_{-1}(-X_3))\) and \(-p_1X_2 \nRightarrow p_{0.6}(-X_2) \nRightarrow p_{-1}(-X_3)\) are fuzzy clauses. \((p_{0.3}X_1 \nRightarrow p_{0.6}(-X_2) \nRightarrow p_{-1}(-X_3))\) is a simple formula. In general, let \(A\) be a formula, write \(A^1 = A\), \(A^0 = \neg A\). Then a fuzzy literal expression can denote as \(p_\alpha X^\varepsilon_1\), where \(\varepsilon, \delta \in \{0, 1\}\). A fuzzy clause can be written as \((p_\alpha X^\varepsilon_i)_1 \nRightarrow \cdots \nRightarrow (p_\alpha X^\varepsilon_i)_n\), where \(\varepsilon_i, \delta_i \in \{0, 1\}, X_i\) are atoms, \(i = 1, \ldots, n\).

**Definition 8.** Let \(X_1, \ldots, X_n; Y_1, \ldots, Y_m\) be atoms. A simple formula \(P \rightarrow C\) is said a local implication formula if atoms appearing in \(P\) belong to \(\{X_1, \ldots, X_n\}\) and atoms appearing in \(C\) belong to \(\{Y_1, \ldots, Y_m\}\). Let \(w_1, \ldots, w_n \in [0, 1]\) and \(w_1 + \cdots + w_n = 1\), \(s_1, \ldots, s_m \in [0, 1]\) and \(s_1 + \cdots + s_m = 1\). Denote \(A = p_{\alpha_1}X_1 \nRightarrow \cdots \nRightarrow p_{\alpha_n}X_n\) and \(B = p_{\beta_1}Y_1 \nRightarrow \cdots \nRightarrow p_{\beta_m}Y_m\). Then the formula \(A \rightarrow B\) is said a whole implication formula.

A correspondence can now be set up between logical statements and linear constraints on the fuzzy variables as follows: Let \(\nu\) be an arbitrary valuation.

1. Suppose \(X\) and \(Y\) are atoms and denote \(x = \nu(X), y = \nu(Y), 0 \leq x, y \leq 1\). Given \(\lambda, \alpha, \beta \in [0, 1]\).
   - (i) \(\models \lambda X\) (means that \(X\) is \(\lambda\)-true) iff \(x - \lambda \geq 0\).
   - (ii) \(\models \lambda X \& Y\) (means that \(X\) and \(Y\) are \(\lambda\)-true) iff \(x + y - \lambda \geq 1\) (i.e., \(x \otimes y \geq \lambda\)).
   - (iii) \(\models \lambda X \nRightarrow Y\) (means that \(X\) or \(Y\) is \(\lambda\)-true) iff \(x + y - \lambda \geq 0\) (i.e., \(x \oplus y \geq \lambda\)).
   - (iv) \(\models \lambda X \rightarrow Y\) (means that \(\text{if } X \text{ then } Y\) is \(\lambda\)-true) iff \(x - y - \lambda \leq 1\) (i.e., \(x \Rightarrow y \geq \lambda\)).
   - (v) \(\models \lambda (p_{\alpha}X)\) iff \(\alpha x - \lambda \geq 0\).
   - (vi) \(\models \lambda (\neg p_{\alpha}X)\) iff \(\alpha x + \lambda \leq 1\).
   - (vii) \(\models \lambda (p_{\alpha}(-X))\) iff \(\alpha x + \lambda \leq \alpha\).
   - (viii) \(\models \lambda (p_{\alpha}(-X))\) iff \(-\alpha x - \lambda \geq 1 - \alpha\).
(ix) \( \vdash_{\lambda} (p_a X)^\lambda \) iff \( (ax)^\lambda - \lambda \geq 0 \) (where \( x^1 = x, x^0 = x^c \)).

(xi) \( \vdash_{\lambda} (p_a X & p_b Y) \) iff \( ax + \beta y - \lambda \geq 0 \) (i.e., \( ax \otimes \beta y \geq \lambda \)).

(xii) \( \vdash_{\lambda} (p_a X \rightarrow p_b Y) \) iff \( ax - \beta y + \lambda \leq 1 \) (i.e., \( ax \Rightarrow \beta y \leq \lambda \)).

Therefore, if \( A \) is a local implication formula, \( \vdash_{\lambda} A \) is equivalent to a linear inequality on the fuzzy variables.

(2) Suppose \( A \) is a fuzzy clause, \( A = (p_a X_{i1}^{\lambda})^{\beta_1} \cdots (p_a X_{in}^{\lambda})^{\beta_n} \), where \( \epsilon_i, \delta_i \in \{0, 1\} \). Denote \( x_i = \nu(X_i) \), then

\[ (xix) \quad \vdash_{\lambda} A \text{ iff } (\alpha_1 x_1)^{\delta_1} + \cdots + (\alpha_n x_n)^{\delta_n} - \lambda \geq 0. \]

Therefore, if \( A \) is a local implication formula, \( \vdash_{\lambda} A \) is equivalent to a linear inequality on the fuzzy variables.

(3) Suppose \( A \) is a fuzzy phrase, \( A = (p_a X_{i1}^{\lambda})^{\beta_1} \cdots (p_a X_{in}^{\lambda})^{\beta_n} \), where \( \epsilon_i, \delta_i \in \{0, 1\} \). Denote \( x_i = \nu(X_i) \), then

\[ (xx) \quad \vdash_{\lambda} A \text{ iff } (\alpha_1 x_1)^{\delta_1} + \cdots + (\alpha_n x_n)^{\delta_n} - \lambda \geq 0. \]

(4) Let \( w_1, \ldots, w_n \in [0, 1] \) and \( w_1 + \cdots + w_n = 1 \); \( s_1, \ldots, s_m \in [0, 1] \) and \( s_1 + \cdots + s_m = 1 \). \( A = p_{w_1} X_1 \& \cdots & p_{w_n} X_n \) and \( B = p_{s_1} Y_1 \& \cdots & p_{s_m} Y_m \). Then

\[ (xxi) \quad \vdash_{\lambda} A \rightarrow B \text{ iff } w_1 x_1 + \cdots + w_n x_n - s_1 y_1 - \cdots - s_m y_m + \lambda \leq 1. \]

Now, we give a main theorem of this section.

**Theorem 7.** Let \( X_1, \ldots, X_n; Y_1, \ldots, Y_m \) be atoms; \( P_i = (p_{a_{ij}} X_{i1}^{\lambda})^{\beta_{ij}} \& \cdots \& (p_{an} X_{in}^{\lambda})^{\beta_{nj}} \), where \( \alpha_{ij}, \epsilon_i, \delta_i \in \{0, 1\} \), for \( i = 1, \ldots, k; j = 1, \ldots, n; C_i = (p_{\beta_i} Y_{i1}^{\eta_i})^{\gamma_{i1}} \cdots (p_{\beta_{ij}} Y_{in}^{\eta_i})^{\gamma_{ijn}} \), where \( \beta_i, \gamma_i, \eta_i \in \{0, 1\} \), for \( i = 1, \ldots, k; j = 1, \ldots, m; P_1 \rightarrow C_1 \) is a local implication formula, for \( i = 1, \ldots, k; w_1, \ldots, w_n \in [0, 1] \) and \( w_1 + \cdots + w_n = 1 \); \( s_1, \ldots, s_m \in [0, 1] \) and \( s_1 + \cdots + s_m = 1 \). Then to find the maximum \( \lambda \in [0, 1] \) such that the whole implication formula \( A \rightarrow B \) is a \( \lambda \)-logical consequence of the local implication formulas \( P_1 \rightarrow C_1, \ldots, P_k \rightarrow C_k \) is equivalent to find \( \lambda = 1 - \text{max}(w_1 x_1 + \cdots + w_n x_n - s_1 y_1 - \cdots - s_m y_m, 0) \), where \( (x_1, \ldots, x_n; y_1, \ldots, y_m) \) is an optimal solution of the following linear programming problem:

\[
\text{(LP)} \quad \text{max } w_1 x_1 + \cdots + w_n x_n - s_1 y_1 - \cdots - s_m y_m \\
\text{s.t. } (\alpha_1 x_1^{\epsilon_1})^{1-\delta_1} + \cdots + (\alpha_n x_n^{\epsilon_n})^{1-\delta_n} + (\beta_1 y_1^{\gamma_1})^{1-\eta_1} + \cdots + (\beta_m y_m^{\gamma_m})^{1-\eta_m} \geq 1, \\
i = 1, \ldots, k; \\
0 \leq x_t \leq 1, t = 1, \ldots, n; \\
0 \leq y_u \leq 1, u = 1, \ldots, m.
\]

**Proof.** By Definition 6, \( (P_1 \rightarrow C_1, \ldots, P_k \rightarrow C_k) \vdash_{\lambda} (A \rightarrow B) \) means: for any valuation \( \nu \) such that \( \nu(P_i \rightarrow C_i) = \cdots = \nu(P_k \rightarrow C_k) = 1, \nu(A \rightarrow B) \geq \lambda \) holds. Let \( x_t = \nu(X_t), y_u = \nu(Y_u), t = 1, \ldots, n; u = 1, \ldots, m \). Clearly, \( x_t, y_u \in [0, 1] \), for \( t = 1, \ldots, n; u = 1, \ldots, m \). Now, \( \nu(P_i \rightarrow C_i) = 1 \) if and only if

\[
(\alpha_1 x_1^{\epsilon_1})^{1-\delta_1} + \cdots + (\alpha_n x_n^{\epsilon_n})^{1-\delta_n} + (\beta_1 y_1^{\gamma_1})^{1-\eta_1} + \cdots + (\beta_m y_m^{\gamma_m})^{1-\eta_m} \geq 1, \quad i = 1, \ldots, k; \tag{\star}
\]

and \( \nu(A \rightarrow B) \geq \lambda \) if and only if

\[
w_1 x_1 + \cdots + w_n x_n - s_1 y_1 - \cdots - s_m y_m + \lambda \leq 1. \tag{\star\star}
\]

Therefore, if \( \lambda \) is the maximum such that the whole implication formula \( A \rightarrow B \) is a \( \lambda \)-logical consequence of the local implication formulas \( P_1 \rightarrow \)
In this section, we apply our approach to an example presented by Castro et al.\textsuperscript{5} Here is their example: Suppose the knowledge base is stated as a set of statements in the propositional calculus:

\begin{align*}
\text{(1)} & \quad \text{If the bond market goes up and the interest rates decrease, either the stock market goes down or taxes are not raised;} \\
\text{(2)} & \quad \text{If the political situation is unstable and the currency is devalued, the inflation increases;} \\
\text{(3)} & \quad \text{If the interest rates increase and the bond market goes down, the stock goes up;} \\
\text{(4)} & \quad \text{If the labor situation is unstable and the political situation is stable, the bond market goes up;} \\
\text{(5)} & \quad \text{If the inflation decreases and the labor situation is unstable, the currency is devalued or political situation is stable;} \\
\text{(6)} & \quad \text{If the stock market goes up, at least one favorable condition exists;} \\
\text{(7)} & \quad \text{If the public deficit increases, and labor situation is unstable, the inflation increases;} \\
\text{(8)} & \quad \text{If the public deficit increases and taxes are raised, the inflation increases;} \\
\text{(9)} & \quad \text{If the political situation is unstable and the employed decrease, the stock market goes down;} \\
\text{(10)} & \quad \text{If the public deficit decreases and taxes are not raised, the interest rate increases;} \\
\text{(11)} & \quad \text{If currency is devalued then the public deficit increases or taxes are raised.}
\end{align*}

We can represent atoms and their negations in Table I. The statements (1)–(11) can be written as the following formulas:

\begin{align*}
A_1 &= (X_2 \& \neg X_5) \rightarrow (\neg X_1 \& Y_3) = (X_1 \& X_2 \& \neg X_5) \rightarrow Y_1; \\
A_2 &= (\neg Y_3 \& \neg X_4) \rightarrow \neg X_3 = (X_3 \& \neg X_4) \rightarrow Y_2;
\end{align*}
Table 1. Atoms and their negations in the example.

<table>
<thead>
<tr>
<th>Economical Evolution</th>
<th>Country Situation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_1$: The stock market goes up.</td>
<td>$Y_1$: Labor situation is stable.</td>
</tr>
<tr>
<td>$\neg X_1$: The stock market goes down.</td>
<td>$\neg Y_1$: Labor situation is unstable.</td>
</tr>
<tr>
<td>$X_2$: The bond market goes up.</td>
<td>$Y_2$: Political situation is stable.</td>
</tr>
<tr>
<td>$\neg X_2$: The bond market goes down.</td>
<td>$\neg Y_2$: Political situation is unstable.</td>
</tr>
<tr>
<td>$X_3$: The inflation decreases.</td>
<td>$Y_3$: Taxes are not raised.</td>
</tr>
<tr>
<td>$\neg X_3$: The inflation increases.</td>
<td>$\neg Y_3$: Taxes are raised.</td>
</tr>
<tr>
<td>$X_4$: Currency is revalued.</td>
<td>$Y_4$: The public deficit decreases.</td>
</tr>
<tr>
<td>$\neg X_4$: Currency is devalued.</td>
<td>$\neg Y_4$: The public deficit increases.</td>
</tr>
<tr>
<td>$X_5$: The interest rates increases.</td>
<td>$Y_5$: The employed increases.</td>
</tr>
<tr>
<td>$\neg X_5$: The interest rates decreases.</td>
<td>$\neg Y_5$: The employed decreases.</td>
</tr>
</tbody>
</table>

A positive economical evolution may be defined as $A = M_5(X_1, X_2, X_3, X_4, X_5)$, so a negative economical evolution will be defined as $\neg A = \neg M_5(X_1, X_2, X_3, X_4, X_5)$. Similarly, a favorable country situation may be defined as $B = M_5(Y_1, Y_2, Y_3, Y_4, Y_5)$ and an unfavorable country situation as $\neg B = \neg M_5(Y_1, Y_2, Y_3, Y_4, Y_5)$. Does a positive economical evolution imply a favorable country situation? In other words, does $A \rightarrow B$?

It should be noted that $A_1, \ldots, A_{11}$ are all local implication formulas. But $A \rightarrow B$ is a whole implication formula. To find the maximum $\lambda$ such that $(A_1, \ldots, A_{11}) \vdash (A \rightarrow B)$, by Theorem 7, is equivalent to solve the following linear programming problem (LP):

$$\begin{align*}
\text{max} & \quad \frac{1}{5}(x_1 + x_2 + x_3 + x_4 + x_5 - y_1 - y_2 - y_3 - y_4 - y_5) \\
\text{s.t.} & \quad -x_1 - x_2 + x_3 + y_3 \geq -1 \\
& \quad -x_3 + x_4 + y_2 \geq 0 \\
& \quad x_1 + x_2 - x_3 \geq 0 \\
& \quad x_2 + y_1 - y_2 \geq 0 \\
& \quad -x_3 - x_4 + y_1 + y_2 \geq -1 \\
& \quad -x_1 + y_1 + y_2 + y_3 + y_4 \geq 0 \\
& \quad -x_3 + y_1 + y_4 \geq 0 \\
& \quad -x_3 + y_3 + y_4 \geq 0 \\
& \quad -x_1 + y_2 + y_5 \geq 0 \\
& \quad x_5 - y_2 - y_4 \geq -1
\end{align*}$$
An optimal solution of this (LP) problem is: $x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1, x_5 = 1, y_1 = y_2 = y_3 = y_4 = y_5 = 0$. The optimal value is 0.6. Let $\lambda = 1 - \max (0.2(x_1 + x_2 + x_3 + x_4 + x_5 - y_1 - y_2 - y_3 - y_4 - y_5), 0) = 0.4$. It follows that "a positive economic evolution implies a favorable country situation" is a 0.4-logical consequence of the knowledge base.

In Castro et al.,\(^5\) the authors applied fuzzy Boolean programming to solve the above problem. They used 20 Boolean variables based on classical two-valued logic. Our approach is based on fuzzy neuron logic and uses only 10 fuzzy variables. The methodology based on fuzzy neural networks and linear programming is more reasonable, because knowledge represented in a knowledge base may not be precise.

V. CONCLUSIONS

In this article, based on the fundamental theorem of fuzzy neural logic, we establish the propositional calculus of fuzzy neural logic. Linear programming approach taken from the propositional calculus of fuzzy neural logic is a useful tool to answer imprecise questions in knowledge-based systems. It seems possible to use network approach to solve logical problems. We will investigate this interesting question in a future paper.

References