NEURAL NETWORKS COMPUTING THE DUNGEAN SEMANTICS OF ARGUMENTATION

Yoshiaki Gotou
Graduate School of Science and Technology, Niigata University, Niigata, Japan

Takeshi Hagiwara, Hajime Sawamura
Institute of Science and Technology, Niigata University, Niigata, Japan

Keywords: Argumentation, Semantics, Neural network.

Abstract: Argumentation is a leading principle foundationally and functionally for agent-oriented computing where reasoning accompanied by communication plays as essential role in agent interaction. In the work of (Makiguchi and Sawamura, 2007a) (Makiguchi and Sawamura, 2007b), they constructed a simple but versatile neural network for the grounded semantics (the least fixed point semantics) in the Dung's abstract argumentation framework (Dung, 1995). This paper further develop its theory so that it can decide which argumentation semantics (admissible, stable, complete semantics) a given set of arguments falls into. In doing so, we construct a more simple but versatile neural network that can compute all extensions of the argumentation semantics. The result leads to a neural-symbolic system for argumentation.

1 INTRODUCTION

Much attention and effort have been devoted to the symbolic argumentation so far (Chesñevar et al., 2000)(Prakken and Vreeswijk, 2002)(Besnard and Doutre, 2004)(Rahwan and Simari, 2009), and its application to agent-oriented computing. Argumentation can be a leading principle both foundationally and functionally for agent-oriented computing where reasoning accompanied by communication plays an essential role in agent interaction. Dung's abstract argumentation framework and argumentation semantics (Dung, 1995) have been one of the most influential works in the area and community of computational argumentation as well as logic programming and non-monotonic reasoning. A. Garcez et al. proposed a novel approach to argumentation, called the neural network argumentation (Garcez et al., 2009). In the papers (Makiguchi and Sawamura, 2007a)(Makiguchi and Sawamura, 2007b), they dramatically developed their initial ideas on the neural network argumentation to various directions in a more mathematically convincing manner. In this paper, we further develop its theory so that it can decide which argumentation semantics (admissible, stable, complete semantics) a given set of arguments falls into. For this purpose, we will construct a more simple but versatile neural network that can compute all extensions of the argumentation semantics (Dung, 1995)(Caminada, 2006), and lead to a neural-symbolic system for argumentation (Levine and Aparicio, 1994)(Garcez et al., 2009).

The paper is organized as follows. In Section 2, we recall what Dung's influential work on the abstract argumentation framework and argumentation semantics is like, for the preparation of the succeeding sections. In Section 3, we introduce a 4-layer neural network and a translation of a given abstract argumentation framework to it, and a calculation of the argumentation semantics with it. In Section 4, we describe an implementation of our neural argumentation with illustrating examples. Final section include previous work,future work, and concluding remarks.

2 ARGUMENTATION SEMANTICS

Argumentation semantics is an important subject since it tells us what correct or justified arguments are. However, there can be plural semantics for argumen-

Goto Y., Hagiwara T. and Sawamura H..
NEURAL NETWORKS COMPUTING THE DUNGEAN SEMANTICS OF ARGUMENTATION.
DOI: 10.5220/0003656600050014
In *Proceedings of the International Conference on Neural Computation Theory and Applications* (NCTA-2011), pages 5-14
ISBN: 978-989-8425-84-3

tation, mainly derived from the nature of argumentation. We introduce the most basic and influential definitions for argumentation semantics which originated with (Dung, 1995).

2.1 Argumentation Framework

The argumentation semantics begins with the definition of (abstract) argumentation framework (Dung, 1995). We are then concerned with calculating all extensions of the semantics for an argumentation framework. Then we assume the argumentation framework to be finite.

Definition 1 (Argumentation Framework) (Dung, 1995). An argumentation framework ($\mathcal{AF}$) is a pair $\langle AR, attacks \rangle$ where AR is a finite set of arguments, and attacks is a binary relation over AR (in symbols, attacks $\subseteq$ AR $\times AR$).

Each element A $\in$ AR is called argument and (A,B) means argument A attacks argument B. An argumentation framework can be represented as a directed graph where the arguments are represented as nodes and the attack relation is represented as arrows. In this paper, we don't consider the internal structure of each of the arguments. For this reason, we don't refer to the structure of attack relations.

Example 1 (Ralph goes Fishing) (Caminada, 2008). Consider the following arguments:

Argument A. Ralph goes fishing because it is Sunday.

Argument B. Ralph does not go fishing because it is Mother's day, so he visits his parents.

Argument C. Ralph cannot visit his parents, because it is a leap year, so they are on vacation.

Three arguments are represented as A, B and C. In this case, we can see (B, A) as an attack relation because B says that Ralph does not go fishing against A. Similarly to (B, A) we can see (C, B) as an attack relation. An example of an argument framework of the argumentation is given in Figure 1.

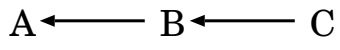


Figure 1: $\mathcal{AF} = \langle \{A,B,C\}, \{(B,A),(C,B)\} \rangle$.

2.2 Dung's Argumentation Semantics

The following definitions are basic notions and used in defining the argumentation semantics.

Definition 2. Let (AR,attacks) be an argumentation framework,A,B$\in$ AR and S$\subseteq$ AR.
$attacks(A,B)$ iff $(A,B) \in attacks$.
$S^+ = \{A \in AR | attacks(S,A)\}$.
$attacks(S,A)$ iff $\exists C \in S(attacks(C,A))$.

In the definition above, S^+ means a set of arguments attacked by the arguments belonging to the S.

Definition 3 (Conflict-free (Dung, 1995)). Let (AR, attacks) be an argumentation framework and let S $\subseteq$ AR. S is said to be conflict-free iff S $\cap$ S^+ = $\emptyset$.

In example 1, according to the definition 3, {A,C} is conflict-free but {A,B} is *not* conflict-free. The notion of conflict-free means that there don't exsit any attack relations each other.

The notion of defend is a core of argumentation semantics and defined as follows.

Definition 4 (Defend (Dung, 1995)). Let (AR,attacks) be an argumentation framework, A $\in$ AR and S $\subseteq$ AR. S is said to defend A (in symbols, defends(S,A)) iff $\forall$B$\in$AR(attacks(B,A)$\rightarrow$attacks(S,B)).

In example 1, according to the definition 4, $defends(\{C\},A)$ holds. And note that there are at least two attack relations when $defends(S,A)$ holds (except S=$\emptyset$). For example, (C,B) and (B,A) exist in example 1. For this reason, the neural network to be proposed in section 3 needs input, first hidden, and second hidden layer. So we can obtain two connections from input layer to second hidden layer. At the same time the neural net work consisted of the 3-layer implements the notion of $defend$.

The following characteristic function F is useful for understanding the argumentation semantics and defined by the notion of $defend$.

Definition 5 (Characteristic Function F) (Dung, 1995). Let (AR,attacks) be an argumentation framework and S $\subseteq$ AR. We introduce a characteristic function as follows:

- $F : 2^{AR} \rightarrow 2^{AR}$
- $F(S) =$ {A$\in$AR | defends(S,A)}

With these in mind, we give a series of definitions for the argumentation semantics.

Definition 6 (Admissible Set (Dung, 1995)). Let (AR,attacks) be an argumentation framework and S $\subseteq$ AR. S is said to be admissible
iff S is conflict-free and $\forall$ A $\in$ AR (A $\in$ S $\rightarrow$ defends(S,A))
iff S is conflict-free and S $\subseteq$ F(S).

According to Definition 6, the empty set is surely admissible set.

Definition 7 (Preferred Extension (Dung, 1995)). Let (AR,attacks) be an argumentation framework and S $\subseteq$ AR. S is said to be a preferred extension iff S is a maximal (w.r.t set-inclusion) admissible set.

Definition 8 (Complete Extension (Dung, 1995)). Let (AR,attacks) be an argumentation framework and S $\subseteq$ AR. S is said to be a complete extension
iff S is conflict-free and $\forall$ A $\in$ AR (A $\in$ S $\leftrightarrow$ defends(S,A))
iff S is conflict-free and S=F(S).

Definition 9 (Stable Extension (Dung, 1995)). Let (AR,attacks) be an argumentation framework and S $\subseteq$ AR. S is said to be a stable extension
iff S is conflict-free and $\forall$ A $\in$ AR (A $\notin$ S$\rightarrow$ attacks(S,A))
iff S is conflict-free and $S \cup S^{+}$=AR.

The definition of the stable extension doesn't need the notion of $defend$. Therefore we need to construct another layer for the neural network in order to compute the stable extension. In fact, output layer is implememted for it.

Definition 10 (Grounded Extension) (Dung, 1995). Let (AR,attacks) be an argumentation framework. The grounded extension is the minimal fixpoint of F.

We represent Admissible Set, Preferred Extension, Complete Extension, Stable Extension and Grounded Extension as AS, PE, CE, SE and GE respectively.

Example 2 (Argumentation Semantics in $\mathcal{AF}$). Let $\langle$\{A,B,C,D,E\},\{(A,B),(B,C),(C,D),(D,C),(D,E),(E,E)\}$\rangle$ be an argumentation framework. The $\mathcal{AF}$ represented as a directed graph is given in figure 2.

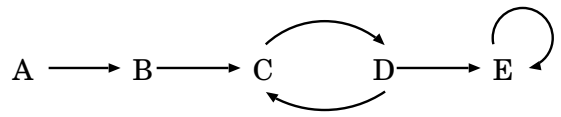


Figure 2: $\mathcal{AF} = \langle \{A,B,C,D,E\}, \{(A,B),(B,C),(C,D),(D,C),(D,E),(E,E)\}\rangle$.

All extensions of the argumentation semantics in $\mathcal{AF}$ are calculated as follows:

- $AS : \emptyset, \{A\}, \{D\}, \{A,C\}, \{A,D\}$
- $CE : \{A\}, \{A,C\}, \{A,D\}$
- $PE : \{A,C\}, \{A,D\}$
- $SE : \{A,D\}$
- $GE : \{A\}$

In the next section, we give a neural network which can computationally accomplish the argumentation semantics described so far.

3 NEURAL NETWORK CALCULATING DUNG'S ARGUMENTATION SEMANTICS

We present how to construct a neural network which computes the argumentation semantics from $\mathcal{AF}$ and determines it using the constructed neural network. Once a neural network has been constructed for an argumentation framework, we can compute its semantics automatically. This means the neural network incorporates such an apparatus as for computing the characteristic function (cf. Definition 5).

From this section, we represent a translated neural network from $\mathcal{AF}$ as $\mathcal{N}$ and let (AR, attacks) be an argumentation framework.

3.1 Translation from $\mathcal{AF}$ to $\mathcal{N}$

3.1.1 The Number of Attacks

Before the translation algorithm from $\mathcal{AF}$ to $\mathcal{N}$, we introduce the function $\mathcal{A}$ which tell us the number of attacks for each argument. And the values generated from the function $\mathcal{A}$ define the threshold θ_{α_k} of the second hidden neuron α_{kh_2}.

Definition 11 (Function $\mathcal{A}$). We introduce a function $\mathcal{A}$ as follows:

- $\mathcal{A}$:AR $\rightarrow \mathbb{N}$
- $\mathcal{A}(X) = |\{(Y,X) | Y \in AR \wedge (Y,X) \in attacks\}|$

Example 3. Examples of function $\mathcal{A}$ in Figure 2 are given as follows:

- $\mathcal{A}(A) = |\emptyset| = 0$
- $\mathcal{A}(B) = |\{(A,B)\}| = 1$
- $\mathcal{A}(C) = |\{(B,C)\},\{(D,C)\}| = 2$
- $\mathcal{A}(D) = |\{(C,D)\}| = 1$
- $\mathcal{A}(E) = |\{(D,E)\},\{(E,E)\}| = 2$

Generally the size of *attacks* equals the sum of function $\mathcal{A}$ for all arguments. For example 3, $attacks = \mathcal{A}(A) + \mathcal{A}(B) + \mathcal{A}(C) + \mathcal{A}(D) + \mathcal{A}(E) = 0+1+2+1+2 = 6$. In addition the value of function

$\mathcal{A}(X)$ means the degree of difficulty of the *defend* for the argument X. Therefore if the function $\mathcal{A}(X)$ takes high value, it is difficult to defend the argument X.

3.1.2 Translation Algorithm

This subsubsection is a core part of this paper and shows the translation algorithm from $\mathcal{AF}$ to $\mathcal{N}$. It is defined as follows:

Step 1 (Input). Given an argumentation framework $\langle AR, attacks\rangle$ where $AR = \{\alpha_1, \alpha_2, ..., \alpha_n\}$.

Step 2 (Types of Neurons). For each argument α_k $\in$AR (1$\leq$k$\leq$n), a neuron is created for each layer (input layer, first hidden layer, second hidden layer and output layer) and named as α_{kX} (see Table 1 about X).

Table 1: Representation rules of neuron.

Layer name of neuron	X
Input layer	i
First hidden layer	h_1
Second hidden layer	h_2
Output layer	o

Step 3 (Weight). Let $\mathbf{a}, \mathbf{b}$ be positive real numbers and satisfy $\sqrt{\mathbf{b}} > \mathbf{a} > 0$. For each argument $\alpha_k \in AR$ (1$\leq$k$\leq$n), do (i) $\sim$ (iii).

- **(i)** connect input neuron α_{ki} to first hidden neuron α_{kh_1} and set $\mathbf{a}$ as the connection weight.
- **(ii)** connect hidden neuron α_{kh_1} to second hidden neuron α_{kh_2} and set $\mathbf{a}$ as the connection weight.
- **(iii)** connect second hidden neuron α_{kh_2} to output neuron α_{ko} and set $\mathbf{a}$ as the connection weight.

Step 4 (Connections and Weights). Let α_k, $\alpha_l$$\in$AR. For each attack relation (α_k, α_l)$\in$ attacks, do (i) $\sim$ (iii).

- **(i)** connect input neuron α_{ki} to first hidden neuron α_{lh_1} and set **-b** as the connection weight.
- **(ii)** connect first hidden neuron α_{kh_1} to second hidden neuron α_{lh_2} and set **-b** as the connection weight.
- **(iii)** connect second hidden neuron α_{kh_2} to output neuron α_{lo} and set -1 as the connection weight.

Step 5 (Threshold). For each argument α_k $\in$AR (1$\leq$k$\leq$n), Set $\mathcal{A}(\alpha_k)\cdot\mathbf{b}$ as the threshold θ_{α_k} of the second hidden neuron α_{kh_2}.

Step 6 (Activation Function of Input Neurons). Set i(x) as the activation function of the input neurons as follows (cf. Figure 3):

$$i(x) = \begin{cases} x & (x \geq 0) \\ 0 & (x < 0) \end{cases} \quad (1)$$

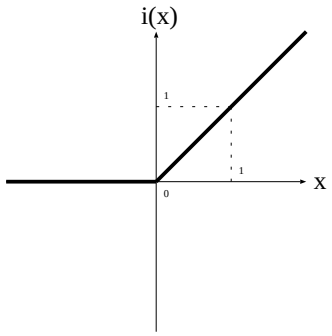


Figure 3: Activation function i(x).

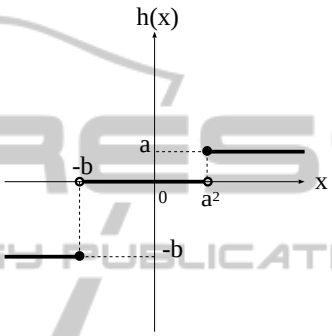


Figure 4: Activation function h(x).

Step 7 (Activation Function of First Hidden Neurons). Set h(x) as the activation function of the first hidden neurons as follows (cf. Figure 4):

$$h(x) = \begin{cases} \mathbf{a} & (x \geq \mathbf{a}^2) \\ 0 & (-\mathbf{b} < x < \mathbf{a}^2) \\ -\mathbf{b} & (x \leq -\mathbf{b}) \end{cases} \quad (2)$$

Step 8 (Activation Function of Second Hidden Neurons). Set g(x) as the activation function of the second hidden neurons as follows (cf. Figure 5):

$$g(x) = \begin{cases} \mathbf{a} & (x \geq \theta_{\alpha_k}) \\ 0 & (x < \theta_{\alpha_k}) \end{cases} \quad (3)$$

Step 9 (Activation Function of Output Neurons). Set f(x) as the activation function of the output neurons as follows (cf. Figure 6):

$$f(x) = \begin{cases} \mathbf{a} & (x \geq \mathbf{a}) \\ 0 & (-\mathbf{a} < x < \mathbf{a}) \\ -\mathbf{a} & (x \leq -\mathbf{a}) \end{cases} \quad (4)$$

All $\mathcal{AF}$ can be translated to $\mathcal{N}$ by the translation algorithm above. Then $\mathcal{N}$ must be a 4-layer neural network. However, the number of neurons is not constant. In detail, the number of neurons equals 4 times the number of the arguments.

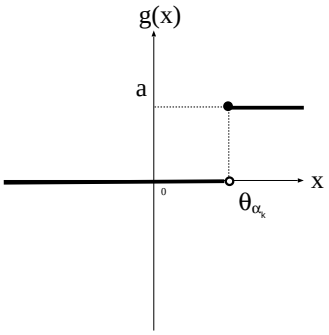


Figure 5: Activation function g(x).

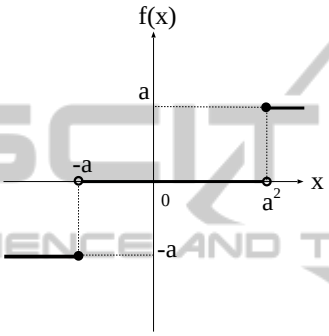


Figure 6: Activation function f(x).

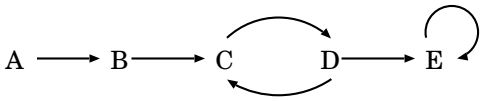
The starting point of the idea of the architectute for constructing the neural network comes from the notion of $defend$. According to the definition 4, at least two attack relations are required for it. Then we can gain an attack relation between two neurons. Similarly, we can gain two attack relations in the three neurons. Threrefore 3-layer neural network can implement the notion of $defend$. And the threshold of the second hidden layer is specified by the number of attack relations and defines the notion of $defend$ and $conflict-free$. For this reason, 3-layer neural network from input layer to second hidden layer can compute the admissible set, complete extension, and grounded extension. However, for the purpose of the computing the stable extension, we add the output layer. Thus we construct the 4-layer neural network in this idea. In addition all settings are required so that the neural network computes argumentation semantics. For this reason, the neural network includes the characterictic function F(cf. Definition 5) and a capability of checking $conflict-free$ (cf. Definition 3).

And the argumentation semantics is computed via checking an input vector converted from a set of arguments (S∈AR) (cf.Definition12) and an output vector which is finally output by $\mathcal{N}$. The criteria for checking the argument semantics is described in 3.3.

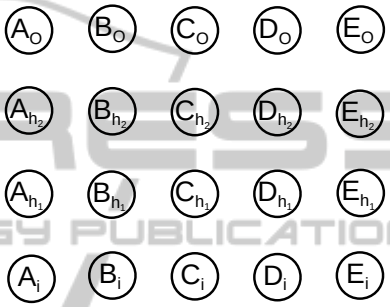
An example of the translation from $\mathcal{AF}$ to $\mathcal{N}$ is seen in Example 4.

Example 4 (Translation from $\mathcal{AF}$ to $\mathcal{N}$). An example of the translation from $\mathcal{AF}$ in Figure 2 to $\mathcal{N}$ is as follow:

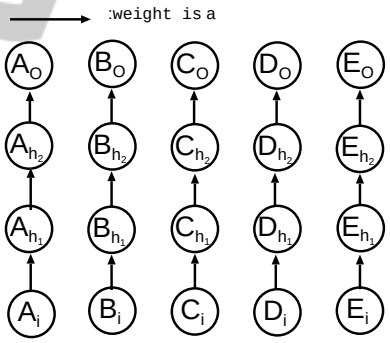
Step 1.



Step 2.



Step 3.



Step 4.

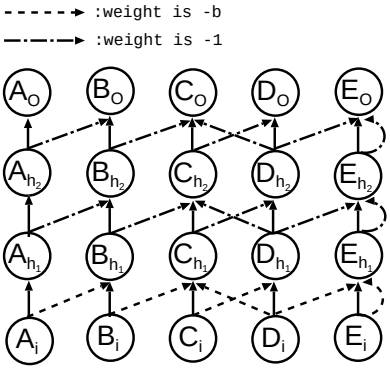


Table 2: Threshold θ_{α_k}

Argument α_k	$\mathcal{A}(\alpha_k)$	Threshold θ_{α_k}
A	0	0
B	1	**b**
C	2	2**b**
D	1	**b**
E	2	2**b**

Step 5. Thresholds for each argument are shown in Table 2.

Step 6 $\sim$ Step 9. Set the activation function to each neuron.

3.2 Calculating Argumentation Semantics by using $\mathcal{N}$

In this subsection, we introduce notions and rules for calculating the argumentation semantics by $\mathcal{N}$.

Notation (Notation of Input-output Values of Neurons). Let $\alpha_k \in AR$. We represent the input values of neurons as $I_{\alpha_{kX}}$ and similarly the output as $O_{\alpha_{kX}}$ (see Table 1 for X). The Figure 7 represents the input-output values of neurons.

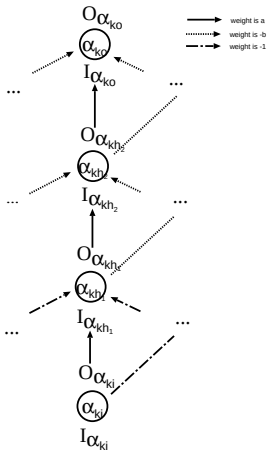


Figure 7: Representation of the input-output values of neurons

An input vector is given to $\mathcal{N}$ at first and used for judging the argumentation semantics. The vector is defined in the following Definition 12.

Definition 12 (First Input Vector $\mathcal{I}$). Let $|AR| = n$ and $S \subseteq AR$. A function $\mathcal{I}$ from a set of arguments to input vector is defined as follws:

- $\mathcal{I} : 2^{AR} \rightarrow \{\mathbf{a}, 0\}^n$
- $\mathcal{I}(S) = [I_{\alpha_{1i}}, I_{\alpha_{2i}}, ..., I_{\alpha_{ni}}]$, $I_{\alpha_{ki}} = \begin{cases} \mathbf{a} & (\alpha_k \in S) \\ 0 & (\alpha_k \notin S) \end{cases}$

Example 5 (First Input Vector). Let Figure 2 be an argumentation framework and S=$\{A,C,E\}$. The first input vector $\mathcal{I}(S) = [\mathbf{a}, 0, \mathbf{a}, 0, \mathbf{a}]$ by Definition 12.

Only when calculating the grounded extension, an output vector is fed back as an input vector until $\mathcal{N}$ gets to be a converging state (cf. Definition 14). The following notions are necessary for the feedbacks.

Definition 13 (Time Round τ). In the computation with a neural network $\mathcal{N}$, the passage of time till the output-vector is read off at the output-neurons since the input-neurons are given an input-vector is called time round, symbolically denoted τ. It has 0 as initial value, and is incremented by 1 every time on input-vector is recurrently given to $\mathcal{N}$.

Definition 14 (Converging State of $\mathcal{N}$). If the input-vector given to $\mathcal{N}$ is identical with the output-vector read off at the same τ, the computation of $\mathcal{N}$ is said to be in a converging state.

3.3 Argumentation Semantics in $\mathcal{N}$

The following definitions show how to interpret a result of computation of $\mathcal{N}$ and determine the argumentation semantics of $\mathcal{N}$. The argumentation semantics in $\mathcal{N}$ can be defined only using the input-output vectors except the grounded extension. Let k be a natural number, $|\ AR\ | = n$, the first input vector be $[i_1, i_2, i_3, ..., i_n]$ and the output vector be $[o_1, o_2, o_3, ..., o_n]$.

Definition 15 (Admissible Set in $\mathcal{N}$). A set S$\subseteq$AR is an admissible set in $\mathcal{N}$ iff $\forall k(i_k = \mathbf{a} \rightarrow o_k = \mathbf{a})$.

Example 6. Let $S = \{D\}$. The input-output vector in the translated $\mathcal{N}$ from Figure 2 (cf. Example 4) is the following:

First Input Vector. $[0, 0, 0, \mathbf{a}, 0]$

Output Vector. $[\mathbf{a}, 0, 0, \mathbf{a}, 0]$

According to Definition 15, $\{D\}$ is an admissible set in $\mathcal{N}$.

Definition 16 (Complete Extension in $\mathcal{N}$). A set S$\subseteq$AR is a complete extension in $\mathcal{N}$ iff $\forall k(i_k = \mathbf{a} \leftrightarrow o_k = \mathbf{a})$.

Example 7. Let $S = \{A, D\}$. The input-output vector in the translated $\mathcal{N}$ from Figure 2 (cf. Example 4) is the following:

First Input Vector. $[\mathbf{a},0,0,\mathbf{a},0]$

Output Vector. $[\mathbf{a},0,0,\mathbf{a},0]$

According to Definition 16, $\{A,D\}$ is a complete extension in $\mathcal{N}$.

Definition 17 (Stable Extension in $\mathcal{N}$). A set S⊆AR is a stable extension in $\mathcal{N}$ iff $\forall k(i_k = \mathbf{a} \leftrightarrow o_k = \mathbf{a}) \wedge \forall l(i_l = 0 \rightarrow o_l = -\mathbf{a})$.

Example 8. Let $S = \{A,C,E\}$. The input-output vector in the translated $\mathcal{N}$ from Figure 2 (cf. Example 4) is the following:

First Input Vector. $[\mathbf{a},0,\mathbf{a},0,\mathbf{a}]$

Output Vector. $[\mathbf{a},-\mathbf{a},\mathbf{a},-\mathbf{a},\mathbf{a}]$

According to Definition 17, $\{A,C,E\}$ is a stable extension in $\mathcal{N}$.

Definition 18 (Grounded Extension in $\mathcal{N}$). Let $S = \emptyset$. $S_g = \{\alpha_k \in \mathrm{AR} \mid O_{\alpha_{ko}}=\mathbf{a}$ in the converging state of $\mathcal{N}\}$ is the grounded extension.

Example 9. The input-output vector in the translated $\mathcal{N}$ from Figure 2 (cf. Example 4) is the following: When calculating the grounded extension, the first input vector must be $\mathcal{I}(\emptyset) = [0,0,0,0,0]$. And the output vector in the converging state of $\mathcal{N}$ is used for checking the grounded extension.

First Input Vector. $[0,0,0,0,0]$

Output Vector. $[\mathbf{a},-\mathbf{a},0,0,0]$

According to Definition 18, $\{A\}$ is a grounded extension in $\mathcal{N}$.

3.4 Soundness of Neural Network Argumentation

The follwing theorem states that the computation of the neural network provides a sound computation in the sense that it yields the same extension as Dung's semantics.

Theorem 1. *Let S ∈ AR. S is an extension of argumentation semantics calculated in $\mathcal{N}$ iff S is an extension of the argumentation semantics in $\mathcal{AF}$.*

Due to the space limitation, we will not describe the details of the proof (see Chapter 5 in (Gotou, 2010)). The idea of the proof is showing the soundness of the definition of the threshold θ_{α_k}. Because the threshold defines the notion of $defend$ and $conflict-free$ and argumentation semantics are mostly defined by them.

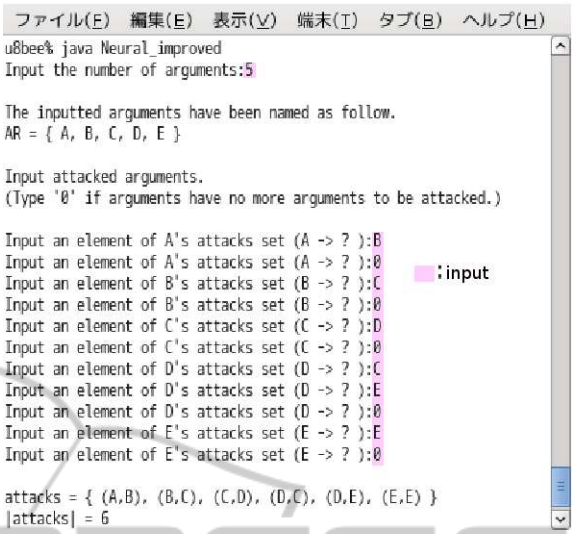


Figure 8: A program behavior from Step 1 to Step 4.

4 IMPLEMENTATION

In this section, we briefly describe an implementation of the neural network argumentation by using its program results. When the program receives $\mathcal{AF}$, it outputs all extensions for each semantics(Admissible, Complete, Stable, Grounded). The following is the flow of the program.

Step 1. Input the number of arguments.

Step 2. Arguments named in alphabetical order are generated.

Step 3. Input the name of an argument which is attacked for each argument, otherwise type'0'.

Step 4. After inputting every counterargument, all attack relations are output.

Step 5. Input-output values of neurons are output for every calculation process.

Step 6. All extensions for each semantics are output.

Here, we will show an example of the program behavior, assuming the program receives $\mathcal{AF} = \langle\{A,B,C,D,E\}, \{(A,B),(B,C),(C,D),(D,C),(D,E),(E,E)\}\rangle$ which is given in Figure 2. An example of the program behavior from Step 1 to Step 4 is given in Figure 8.

Now we calculate some semantics for $\mathcal{AF}$. There are 5 arguments and therefore we input 5 (Step 1). Thereupon 5 arguments (named A, B, C, D, E) are generated (Step 2). Then we input the name of an argument which is attacked. For the argument A in this example, A attacks B and has no more arguments it attacks. Accordingly, we input B at first and next, input

Figure 9: Computation process (1).

0 (Step 3). After inputting every counterargument, all attack relations are outputted (Step 4).

We set 1 as **a** and 2 as **b** in the implementation. The following shows the computation process for checking the grounded extension (input S=$\emptyset$). See Figure 9. The vectors in lines represent the input-output values of each neuron and are listed in alphabetical order about arguments and the top is the input and the bottom is the output. Calculating only the grounded semantics, the output vector is fed back as a new input vector. In the time round 1, the input vector equals to the output vector. We can regard it as a converging state of $\mathcal{N}$. Thus $\{A\}$ is the grounded extension in $\mathcal{N}$ by Definition 18.

The computation process for input $S = \{A,D\}$ is shown in Figure 10. According to Definition 15, 16

Figure 10: Computation process (2).

and 17, $\{A,D\}$ are admissible, complete and stable.

The results of all extensions of the argumentation semantics is shown as Figure 11. We need the power-of-AR computations in $\mathcal{N}$ in order to get all extensions of the argumentation semantics.

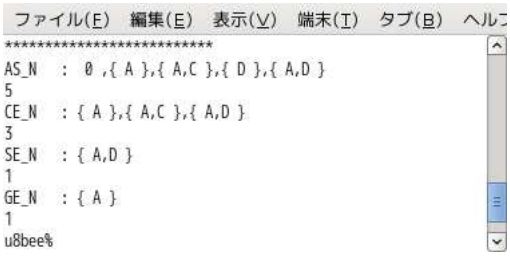


Figure 11: Results of argumentation semantics.

5 PREVIOUS WORK

Previously, Makiguchi and Sawamura proposed the neural network that computes the Dungean argumentation semantics (Makiguchi and Sawamura, 2007a)(Makiguchi and Sawamura, 2007b). However, it turned out that the neural network cannot compute the Dungean argumentation semantics in some abstract argumentation framework. For example, let be $\langle \{A,B,C\},\{(B,A),(C,B)\} \rangle$ an argumentation framework shown in Figure 1 and $S = \{A,C\}$. According to the study (Makiguchi and Sawamura, 2007a)(Makiguchi and Sawamura, 2007b), first input vector =$[1,0,1]$ and input vector =$[0,-1,1]$.This means that $\{A,C\}$ is not justified in admissible set by the work (Makiguchi and Sawamura, 2007a)(Makiguchi and Sawamura, 2007b). However, according to the definition 6, $\{A,C\}$ should be justified in admissible set. Thus the neural network cannot compute some extensions correctly. On the other hand the neural network we proposed can compute all extensions. In the same example, first input vector =$[\mathbf{a},0,\mathbf{a}]$ and input vector =$[\mathbf{a},-\mathbf{a},\mathbf{a}]$. According to the definition 15, $\{A,C\}$ is justified in admissible set as well as Dungean argumentation semantics. For this reason, there are several differences between our work and the work (Makiguchi and Sawamura, 2007a)(Makiguchi and Sawamura, 2007b). And the main differences are shown as follws:

- The former architecture is 3-layer neural network but the latter is 4-layer.
- The former neural network is completely defined by invariable but the latter is mostly defined by variable.
- The former neural network needs the definition of types of attack relation but the latter doesn't.
- The former is the recurrent neural network but the latter is the feedforward neural network (except the computation of the grounded extension).
- The former neural network cannot compute some

extensions but the latter can compute all extensions.

The neural network we proposed is constructed so that it can compute the Dungean argumentation semantics correctly. And the Dungean argumentation semantics is defined by the notions of defend and conflict-free mainly. For this reason, all parameters are set up so that the notions are implememted into the neural network.

6 RELATED WORK

D'Avila Garcez et al. initiated a novel approach to argumentation, called the neural network argumentation (Garcez et al., 2005). However, the semantic analysis for it is missing there. That is, it is not clear what they calculate by their neural network argumentation.

Besnard and Doutre proposed three symbolic approaches to checking the acceptability of a set of arguments (Besnard and Doutre, 2004), in which not all of the Dungean semantics can be dealt with. So it may be fair to say that our approach with the neural network is more powerful than Besnard et al.'s methods.

Vreeswijk and Prakken proposed a dialectical proof theory for the preferred semantics (Vreeswijk and Prakken, 2000). It is similar to that for the grounded semantics (Prakken and Sartor, 1997), and hence can be simulated in our neural network as well.

Hölldobler and his colleagues gave a method to encode a logic program to a 3-layer recurrent neural network and compute the least fixed point of it (Hölldobler and Kalinke, 1994) in the semantics of logic programming. We, on the other hand, constructed a 4-layer neural network, but our neural network computes not only the least fixed point (grounded semantics) but also the fixed points (complete extension) in the argumentation semantics.

7 CONCLUDING REMARKS

It is a long time since connectionism appeared as an alternative movement in cognitive science or computing science which hopes to explain human intelligence or soft information processing. It has been a matter of hot debate how and to what extent the connectionism paradigm constitutes a challenge to classicism or symbolic AI. On the other hand, much effort has been devoted to a fusion or hybridization of neural net computation and symbolic one (Jagota et al., 1999).

In this paper, we proposed a neural network which computes all extensions of an argumentation semantics exactly for every argumentation framework and showed the soundness of neural network argumentation. The results yield a strong evidence to show that such a symbolic cognitive phenomenon as human argumentation can be captured within an artificial neural network.

With the neural argumentation framework presented in the paper, we showed that symbolic dialectical proof theories can be obtained from the neural network computing various argumentation semantics, which allow to extract or generate symbolic dialogues from the neural network computation under various argumentation semantics. The results illustrate that there can exist an equal bidirectional relationship between the connectionism and symbolism in the area of computational argumentation (Gotou et al., 2011).

The simplicity and efficiency of our neural network may be favorable to our future plan such as introducing learning mechanism into the neural network argumentation, implementing the neural network engine for argumentation, and so on. Specifically, it might be possible to take into account the so-called core method developed in (Hölldobler and Kalinke, 1994) and CLIP in (Garcez et al., 2009) although our neural-symbolic system for argumentation is much more complicated due to the complexities and varieties of the argumentation semantics.

The neural argumentation framework allows to build an Integrated Argumentation Environment (IAE) based on the logic of multiple-valued argumentation (Takahashi and Sawamura, 2004), powered by the neural network. We are planning to incorporate the neural argumentation framework to IAE.

REFERENCES

Besnard, P. and Doutre, S. (2004). Checking the acceptability of a set of arguments. In *10th International Workshop on Non-Monotonic Reasoning (NMR 2004*, pages 59–64.

Caminada, M. (2006). Semi-stable semantics. In Dunne, P. E. and Bench-Capon, T. J. M., editors, *Computational models of argument: proceedings of COMMA 2006*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 121–130. IOS Press.

Caminada, M. (2008). A gentle introduction to argumentation semantics. http://icr.uni.lu/caminada/publications/Semantics_Introduction.pdf.

Chesñevar, C. I., Maguitman, G., and Loui, R. P. (2000). Logical models of argument. *ACM Computing Surveys*, 32:337–383.

Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logics programming and n-person games. *Artificial Intelligence*, 77:321–357.

Garcez, A. S. D., Gabbay, D. M., and Lamb, L. C. (2005). Value-based argumentation frameworks as neural-symbolic learning systems. *Journal of Logic and Computation*, 15(6):1041–1058.

Garcez, A. S. D., Lamb, L. C., and Gabbay, D. M. (2009). *Neural-symbolic cognitive reasoning*. Springer.

Gotou, Y. (2010). Neural networks calculating Dung's argumentation semantics. http://www.cs.ie.niigata-u.ac.jp/Paper/Storage/graduation_thesis_gotou.pdf.

Gotou, Y., Hagiwara, T., and Sawamura, H. (2011). Extracting argumentative dialogues from the neural network that computes the dungean argumentation semantics. In *7th International Workshop on Neural-Symbolic Learning and Reasoning (NeSy11)*. To be presented at NeSy11.

Hölldobler, S. and Kalinke, Y. (1994). Toward a new massively parallel computational model for logic programming. In *Proc. of the Workshop on Combining Symbolic and Connectionist Processing, ECAL 1994*, pages 68–77.

Jagota, A., Plate, T., Shastri, L., and Sun, R. (1999). Connectionist symbol processing: Dead or alive? *Neural Computing Surveys*, 2:1–40.

Levine, D. and Aparicio, M. (1994). *Neural networks for knowledge representation and inference*. LEA.

Makiguchi, W. and Sawamura, H. (2007a). A hybrid argumentation of symbolic and neural net argumentation (part i). In *Argumentation in Multi-Agent Systems, 4th International Workshop, ArgMAS 2007, Revised Selected and Invited Papers*, volume 4946 of *Lecture Notes in Computer Science*, pages 197–215. Springer.

Makiguchi, W. and Sawamura, H. (2007b). A hybrid argumentation of symbolic and neural net argumentation (part ii). In *Argumentation in Multi-Agent Systems, 4th International Workshop, ArgMAS 2007, Revised Selected and Invited Papers*, volume 4946 of *Lecture Notes in Computer Science*, pages 216–233. Springer.

Prakken, H. and Sartor, G. (1997). Argument-based extended logic programming with defeasible priorities. *J. of Applied Non-Classical Logics*, 7(1):25–75.

Prakken, H. and Vreeswijk, G. (2002). Logical systems for defeasible argumentation. In *In D. Gabbay and F. Guenther, editors, Handbook of Philosophical Logic*, pages 219–318. Kluwer.

Rahwan, I. and Simari, G. R. E. (2009). *Argumentation in artificial intelligence*. Springer.

Takahashi, T. and Sawamura, H. (2004). A logic of multiple-valued argumentation. In *Proceedings of the third international joint conference on Autonomous Agents and Multi Agent Systems (AAMAS'2004)*, pages 800–807. ACM.

Vreeswijk, G. A. W. and Prakken, H. (2000). Credulous and sceptical argument games for preferred semantics. *Lecture Notes in Computer Science*, 1919:239–??