ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 281-286

Neural assembly binding in linguistic representation

Frank van der Velde & Marc de Kamps

Cognitive Psychology Unit, University of Leiden, Wassenaarseweg 52,
2333 AK Leiden, The Netherlands, vdvelde@fsw.leidenuniv.nl

Abstract. We present a neural architecture of sentence representation. Words are represented with neural cell assemblies. Relations between words are represented with 'structure' assemblies. Word and structure assemblies are bound temporarily to form a sentence representation. We show how multiple sentences can be represented simultaneously, and we simulate how specific information can be retrieved from the architecture. The assemblies are simulated as populations of spiking neurons, in terms of the average firing rate of the neurons in the population.

1. Introduction

We present a neural model of basic sentence structure. Words are represented with neural assemblies. Relations between the words in a sentence cannot be represented with direct associations between these word assemblies. For instance, the association of *mouse-chases-cat* does not distinguish between the sentences *The mouse chases the cat* and *The cat chases the mouse*. Therefore, word assemblies are embedded in a neural architecture in which structural relations can be formed between the word assemblies. The neural architecture is implemented by means of 'structure' assemblies that interact with the word assemblies. The structure assemblies provide the possibility to represent different instantiations of the same word assembly, and they are used to represent elements of syntactic structures. Here, we use structure assemblies for representation of Noun Phrases (NPs) and Verb Phrases (VPs).

Figure 1 presents the representation of the sentence *The mouse chases the cat* in this architecture. It consists of word assemblies, structure assemblies for NPs and VPs, gating circuits used for dynamic control, and memory circuits used to bind word and structure assemblies into a (temporal) representation of the sentence. Thus, temporarily, *mouse*, *chases* and *cat* are bound to different structure assemblies, which in turn are bound to represent the sentence. Structure assemblies are composed of a main assembly (N_i for NP and V_i for VP) and subassemblies, here for agent (a) and theme (t). Subassemblies are connected to the main assembly by gating circuits. Word and structure assemblies are bound by activating memory circuits that connect them. Structure assemblies are bound by activating the memory circuits that connect their agent/theme subassemblies. Similar representations can be formed for sentences like *The cat chases the mouse* and *The mouse sees the dog* (figure 1).

The words *mouse* and *chases* occur in more than one sentence in figure 1. This creates the problem of the multiple instantiation of the assemblies for *mouse* and *chases* [1]. Figure 1 illustrates that the problem of multiple instantiation is solved by binding each word assembly (temporarily) to a unique structure assembly. For instance, the word assembly for *mouse* is bound to the NP assemblies N_1, N_4 and N_5

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 281-286

in figure 1. In this way, *mouse* can be represented as agent in one sentence (by N_1 or N_5) and as theme in another (by N_4). Similarly, the different VP assemblies (V_1 and V_2) represent *chases* in different sentences. The internal structure of the NP and VP assemblies, given by the gating circuits, is of crucial importance in this respect. Without this internal structure, the representations presented in figure 1 would also consist of direct associations between neural assemblies, which would result in a failure to distinguish between *The mouse chases the cat* and *The cat chases the mouse*. With the control of activation provided by gating circuits, the representations of these two sentences can be selectively (re)activated. We will illustrate this in the last section. In particular, we will investigate how information can be retrieved (i.e., answers to binding questions can be produced) in the architecture presented in figure 1, even with multiple instantiation of representations as illustrated in figure 1. First, however, we will describe the gating and memory circuits.

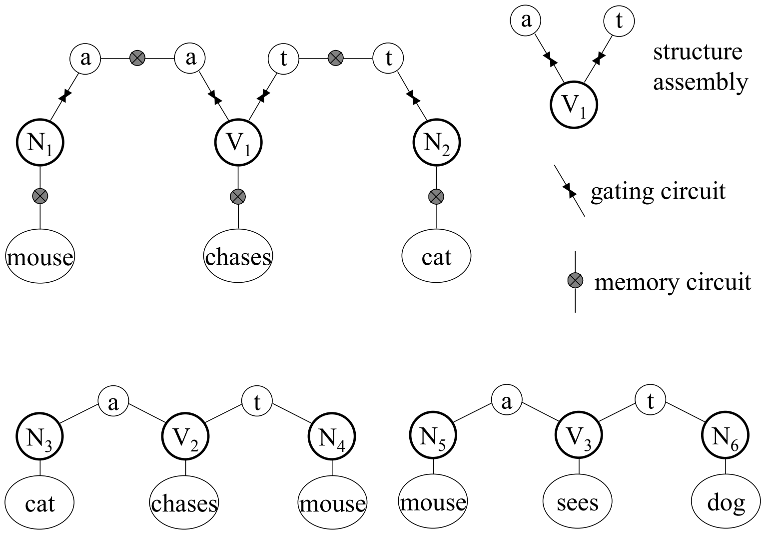

Figure 1. Sentence representation with neural assemblies. Circles and ovals represent populations of neurons (assemblies). V = verb phrase, N = noun phrase, a = agent, t = theme.

2. Gating and memory circuits

Figure 2 (left) illustrates the gating circuit. The overall circuit is in fact a combination of two gating circuits, one for each direction. They are disinhibition circuits [2] that control the flow of activation between two assemblies (X and Y in figure 2) by means of an external control signal. The gating circuit that controls the flow of activation from X to Y operates in the following manner. If the assembly X is active, it activates an inhibition neuron (or group of neurons) i_x, which inhibits the flow of activation from X to X_{out}. When i_x is inhibited by another inhibition neuron (I_x) that is activated by an external 'control signal', X activates X_{out}. In turn, X_{out} activates Y. The gating circuit from Y to X operates in a similar manner.

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 281-286

The memory circuit is presented in figure 2 (right). It also consists of two gating circuits that control the flow of activation from *X* to *Y* and vice versa. In this case, however, the control signal in both gating circuits results from a 'delay assembly'. The delay assembly is activated when *X* and *Y* are active simultaneously, and it remains active for a while due to the reverberating activity in this assembly (see appendix). The memory circuits in figure 1 are active.

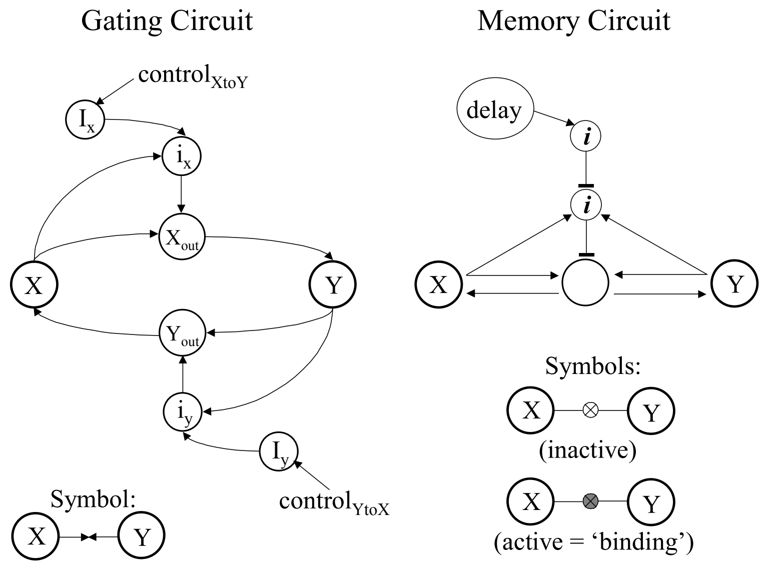

Figure 2: Left: gating circuit. Right: memory circuit. Circles and ovals represent assemblies. Circles with I or i represent inhibitory (populations of) neurons.

3. Retrieving information from the architecture

We will illustrate the ability to retrieve information from this architecture by analyzing and simulating the production of the answer to the question "Whom does the mouse chase?", when the sentences presented in figure 1 are stored simultaneously. The assemblies were simulated as populations of spiking neurons (see the appendix). The simulations are illustrated in the figure 3. The figure (middle) also shows two 'free' VP main assemblies (V_4 and V_5), not used in any sentence representation, to compare the activation of free assemblies with bound assemblies in this process. The vertical lines are used to compare the timing of events. The simulations start at t = 0 ms. Before that time, the only active assemblies are the delay assemblies in the memory circuits (see figure 1).

The question "Whom does the mouse chase?" provides information that *mouse* is the agent of *chases* and it asks for the theme of the sentence *mouse chases x*. The production of the answer consists of the selective activation of the word assembly for *cat*. Backtracking, this requires the selective activation of the main assembly N_2, the theme subassemblies for N_2 and V_1, and the main assembly V_1 (in reversed order). This process proceeds as follows. First, we assume that the question temporarily

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 281-286

activates the representations for *mouse* and *chases* and produces the control signal that activates the gating circuits for the agent subassemblies of the NP assemblies. Figure 1 shows the activation of the assemblies for *mouse* and *chases* (beginning at t = 0 ms). To produce the selective activation of the word assembly for *cat* later on, other word assemblies cannot be active at that moment. Therefore, it is assumed that the word assemblies are inhibited after a certain time, and remain inhibited until *cat* is to be activated. The horizontal bar in figure 1 (right) indicates the time interval in which the word assemblies (*mouse* and *chases*) are active. The end of the interval (at t = 400 ms) is marked by a solid vertical line.

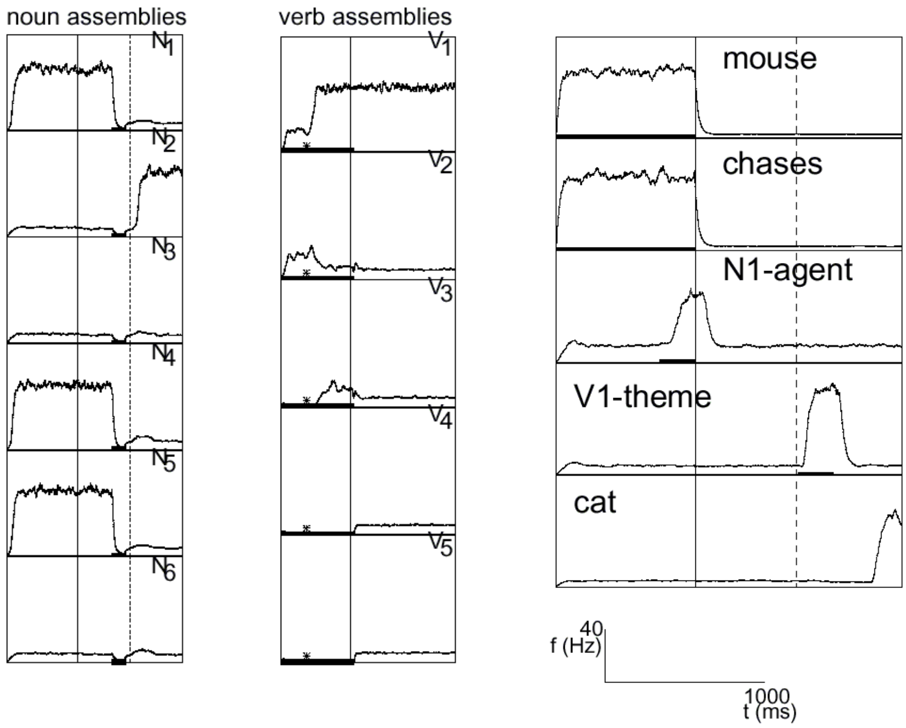

Figure 3. Activation of the neural assemblies in figure 1 (in Hz/ms). Left panel: The noun assemblies N_1 to N_6. Middle panel: The verb assemblies V_1 to V_5. Right panel: The word assemblies for *mouse*, *cat* and *chases*, and the structure assemblies for N_1-agent and V_1-theme.

The activation of *mouse* results in the activation of N_1, N_4, and N_5, and the activation of *chases* results in the activation of V_1 and V_2 (figure 3). As indicated with the solid vertical line in figure 3, N_1, N_4, and N_5 remain active when *mouse* is inhibited. This results from the reverberating ('delay') properties of main assemblies (see the appendix). As long as V_1 and V_2 are both active, the question "Whom does the mouse chase?" cannot be answered. To produce the answer, the gating circuits for the theme VP subassemblies have to be activated, because the question asks for the theme of *mouse chases x*. However, when both V_1 and V_2 are active, this will result in the activation of the theme subassemblies for V_1 and V_2, and, in turn, of *cat* and *mouse* (via N_2 and N_4). To prevent this, a WTA competition between V_1 and V_2 has to occur, with V_1 as the winner.

The competition process between the VP assemblies proceeds as follows. VP main assemblies are connected to a population of inhibitory neurons. In comparison with the NP assemblies activated by *mouse* (figure 3, left), the activity of V_1 and V_2 (figure 3, middle) , initiated by *chases*, is reduced due to the competition between the VP assemblies. The competition can be decided by activating the gating circuits for the agent subassemblies. This results in the activation of the agent subassemblies for N_1, N_4 and N_5, because they are the active NP assemblies (figure 3, left). The activation of the N_1 agent subassembly is illustrated in figure 3 (right). The horizontal bar here indicates the time interval in which the gating circuits are activated (from t = 150 ms to t = 400 ms). The beginning of this interval is indicated by the asterix in figure 3 (middle). The active agent subassemblies N_1 and N_5 are bound to the VP assemblies V_1 and V_3 respectively (see figure 1). Thus, the VP assemblies V_1 and V_3 receive activation from these NP assemblies when the 'agent' gating circuits are activated. (The agent subassembly of N_4 is not bound to a VP assembly, because N_4 is bound to a VP assembly with its theme subassembly, see figure 1). As a result, V_1 wins the competition between the VP assemblies, because V_1 receives activation from *chases* and N_1, whereas V_2 only receives activation from *chases,* and V_3 only receives activation from N_5. Figure 3 (middle) shows that V_1 is the only active VP assembly after this competition process. The activation of V_2 and V_3 is reduced to the level of the 'free' assemblies V_4 and V_5.

When V_1 remains as the only active VP assembly, the answer *cat* can be produced by activating the theme gating circuits in the direction from VP to NP. This will produce the selective activation of N_2, which is the NP assembly bound to *cat* in figure 1, provided that the active NP main assemblies (N_1, N_4 and N_5 in figure 3) are inhibited first. The horizontal bar in figure 3 (left) illustrates the time interval of this inhibition (from t = 600 ms to t = 650 ms). After the inhibition of the active NP assemblies, the theme gating circuits can be activated. The horizontal bar in figure 3 (V1-theme) illustrates the time interval (from t = 700 ms to t = 800 ms). The onset of this event is also illustrated by the dashed vertical line in figure 3 (left, right). As a result, the theme subassembly of V_1 and the main assembly N_2 are now selectively activated as well. As a result, the word assembly for *cat* can be activated.

References

1. Jackendoff, R. (2002). Foundations of Language. Oxford: Oxford University Press.
2. Van der Velde, F. & de Kamps, M. (2001). From knowing what to knowing where: Modeling object-based attention with feedback disinhibition of activation. Journal of Cognitive Neuroscience, 13, (4), 479-491.
3. Gerstner, W. (1995). Time structure of the activity in neural-network models. Physical Review E, 51, 738-758.

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 281-286

Appendix

The simulations are based on a network of excitatory and inhibitory populations [3]. The population rate A_i is given by (with $\alpha = E$ for excitatory populations and $\alpha = I$ for inhibitory populations):

$$\tau_\alpha \frac{dA_i}{dt} = (-A_x + F(input_A \Sigma_j w_{ij} A_{ij})) + A_i I_{noise} \tag{1}$$

τ_E (τ_I) is the time constant, with $\tau_E = 10$ ms and $\tau_I = 5$ ms. The w_{ij} (or $w_{j \to i}$) are the efficacies from population j onto population i: w_{ij} is negative iff j is an inhibitory population. Every 1 ms, a fraction I_{noise} of the population activation is injected into each population with $\mu = 0, \sigma = 0.02$.
For $F(x)$ we took:

$$F(x) = \frac{f_{max}}{(1 + e^{-\beta(x-\theta)})} \tag{2}$$

with $f_{max} = 30$ Hz, $\beta = 1$ and $\theta = 3$.
The gating circuits in our simulation resulted from (1) by inserting X, Y, X_{out}, Y_{out}, i_x, i_y, I_x, and I_y in (1) in line with the diagram of figure 1 (left). We took $w_{X \to X_{out}} = w_{Y \to Y_{out}} = w_{X \to i_x} = w_{Y \to i_y} = 0.25$ and $w_{i_x \to X_{out}} = w_{i_y \to Y_{out}} = w_{I_x \to i_x} = w_{I_y \to i_y} = 1$. We took $w_{Y_{out} \to X} = w_{X_{out} \to Y} = 0.1$. The gating circuit can be activated by the input signals $control_{XtoY}$ and $control_{YtoX}$, from two outside populations, with activation f_{max} and $w_{control} = 0.2$. The memory circuits were simulated as gating circuits, with control signal 0 ('off') or f_{max} ('on'), and $w_{Y_{out} \to X} = w_{X_{out} \to Y} = 0.2$.
We assumed the following properties for a delay population: 1. It is active once its input has been above a threshold θ_{delay} in the past and it has not been deactivated since. 2. It is deactivated once the net afferents to the assembly passes a certain negative threshold θ_{deact} (i.e., there is net inhibition). 3. If it is inactive, it functions as an ordinary population of excitatory neurons.
To treat a delay population as part of the network, we assumed: 1. If its activity is above threshold θ_{delay} and net input is excitatory, then its time constant is $\tau_{delay} = \tau_E$. 2. If its activity is above threshold θ_{delay} but decreasing, while net input is above θ_{deact}, the time constant is very large: τ_{inf}. 3. If net input is below θ_{deact} the time constant is reset to τ_E and, since net input is negative, memory activity will decay within approximately τ_E ms. We took $\theta_{deact} = -0.2$, $\theta_{delay} = 4$ and $\tau_{delay} = 10000$ ms.
Structure assemblies consist of main assemblies and subassemblies. Main assemblies are delay populations. Word assemblies are active with activation given by f_{max} and fraction I_{noise}. They act on stucture assemblies with efficacy $w_{input} = 0.2$. VP structure assemblies are connected with a central inhibitory pool, which acts on them with efficacy $w_{pool \to VP} = 0.03$. The inhibitory pool receives input from the VP assemblies with efficacy $w_{VP \to pool} = 0.03$.
In all, the simulated model consisted of 624 populations. Integration of the system of equations (1) evolved simultaneously for the entire model, using fourth-order Runge-Kutta integration with an integration time step $h = 0.01$ ms.