Vocabulary Learning and Instruction
Volume 11, Issue 1, December 2022
http://vli-journal.org

Vocabulary Networks Workshop 1: Introduction

Paul Meara[a] and Imma Miralpeix[b]
[a]*Swansea University;* [b]*Universitat de Barcelona*

Abstract

The idea that a vocabulary is a network of words is one that has become a common theme of the second language (L2) vocabulary research literature. However, not many people have considered the wider implications of this powerful metaphor. This paper is the first in a series of workshops that examines some of these implications. In this first workshop, we introduce some very simple computational models which illustrate some basic properties of network structures.

The workshop consists of a set of interactive programs that model a vocabulary as a *Cellular Automaton*—a minimally organized network where each word is linked to a small number of other words, and words change their activity status depending on the inputs they receive from other words. These networks exhibit some surprising *emergent properties* which have important implications for the way we understand and think about real vocabulary networks.

Keywords: Cellular automata, Emergent properties, Network structures, Simulations, Vocabulary modeling

1 Introduction

The idea that a vocabulary consists of a set of words that are linked together in a network is one that has become pervasive in the second language (L2) vocabulary research literature. However, not many people have looked at the implications of a networked vocabulary, or considered how a network might be different from a mere heap of words that are not connected to each other. Despite Aitchison's assertion that "words cannot be heaped up randomly in mind" (Aitchison, 2012:5), most discussions of vocabulary networks in the L2 research literature are not much more than a vague metaphor. Occasionally, we find a reference to Collins and Quillian's idea that a vocabulary network might work on "spreading activation": when a word is activated it also activates other words that it is connected to, and this causes a ripple of activation to spread through the entire lexicon (Collins & Loftus, 1975). This is an attractive idea, but, in the L2 literature at least, the details of the mechanism are rarely specified, and this makes this research difficult to evaluate.

Most L2 vocabulary research assumes that the "network" which underpins a vocabulary is basically the same as the networks that we find in word association studies. Again, this looks like a reasonable assumption at first sight, but word association studies tend to use relatively small stimulus sets, and it is not easy to see how a network built out of the responses to, say, 50 stimulus words might scale up to a much larger vocabulary of several thousand words.

Please cite this article as: Meara, P. M. and Miralpeix, I. (2022). Vocabulary networks workshop 1: Introduction. *Vocabulary Learning and Instruction, 11*(1), 17–34. https://doi.org/10.7820/vli. v11.1.meara1

This workshop is an attempt to address these issues. The approach taken in the workshop is a rather unusual one, in that it does not involve collecting empirical data from real language learners. Instead, it uses **simulations**—computer models of vocabulary-like networks which look as though they might provide useful analogues for real vocabularies. This approach turns out to be a very useful tool for L2 vocabulary researchers (e.g., Segalowitz & Bernstein, 1997): its main advantage is that it removes some of the messiness that we usually find in ordinary empirical studies, and allows us to examine in detail some of the assumptions that most of us take for granted. Pulling apart ideas that we usually do not question is almost always a good thing for researchers to do, as it helps to rethink ideas about L2 vocabulary acquisition. In addition, as we shall see, working with simulations is an extraordinarily effective way of generating interesting speculations about how L2 vocabulary networks function in real life.

2 How the Workshop is Structured

The workshop is organized around a series of questions of increasing complexity. We will be examining the questions by working with some specially prepared computer programs that are available on the workshop website.

The workshop is mainly interested in L2 vocabulary networks, but we will start off with a basic introduction to building vocabulary models and running simulations. To start with, we introduce you to a minimal type of network that will serve as a preliminary model of how a vocabulary network might work. Networks of this type are deceptively simple, but they turn out to have some interesting emergent properties that may have important implications for our understanding of real L2 vocabulary networks. Then, we will progressively work with more complex models with several parameters that you can change, allowing you to interact with the network in many different ways. After that, we will focus on specific aspects that can be studied using model networks: vocabulary attrition and loss, bilingual and multilingual lexicons and how they interact with each other, and how we might grow a vocabulary from scratch.

Simulation research has had something of a bad press in the Applied Linguistics research literature, on the grounds that it is so far removed from "the real world" that it is just irrelevant. This workshop is designed to show just how short-sighted this view is. We hope that running these simulations will teach you a very different way of asking questions about vocabulary, and that your own research will be enriched as a result.

3 Part One: First Simulations

3.1 Basic Networks

This section will provide some essential background to a set of simple network models called ***Cellular Automata.*** These models usually consist of a set of objects that interact with each other in some way. Modeling a vocabulary in this way looks as though it should be fairly straightforward. Each word in the

Figure 1. Network 1. A Really Simple Network.

vocabulary is an object that is linked to a number of other words, and all we need to do in order to set up a basic model is to be specific about the connections and how they allow interactions between the words. We will start off by looking at some very small model vocabularies so that you can familiarize yourself with the type questions we will be asking in later parts of the workshop, when we look in more detail at larger, more realistic vocabularies.

We can illustrate the basic approach with the simple network shown in Figure 1.

In this network, we have five words. Each of the words can be in one of two activation states, conventionally called ON and OFF. In addition, each word is connected to one other word. These connections (shown by the arrows), are fairly limited in what they can do: the connection from WordA for instance simply sends a message from WordA to WordB that tells WordB whether WordA is ON or OFF (we call this message "input"), and the connection from WordB just sends a message to WordC that tells WordC whether WordB is ON or OFF. And so on. WordE does not have any outgoing connections.

In Figure 1 all the words start out in the OFF state, so the network does not actually do anything. But we can make this network more interesting by specifying how each word responds to its input. Let us say that if a word receives a message telling it that its input is ON, then it too will go ON. In Figure 1, if WordA is briefly turned ON, then WordB "knows" this, and will turn itself ON, sending a signal to WordC as it does so. Basically, in this network, if a word receives an input from its neighbor, it becomes activated and goes ON. If a word is not receiving an input, it goes OFF.

Now consider what this network does if we briefly activate WordA: WordA will go ON, and because WordA is ON, it sends a signal to WordB. This has the effect of turning WordB ON. Meanwhile, WordA is not receiving an input anymore, so it goes OFF. We now have one activated Word, WordB, and this word sends a signal to WordC, so WordC turns itself ON. WordB meanwhile turns itself OFF. WordC sends a signal to WordD, and turns itself OFF. This signal turns WordD ON, and sends a signal to WordE. WordE turns itself ON briefly, but does not do anything else. WordD will have turned itself OFF, so WordE is no longer getting a signal and will turn itself OFF. In this way, activating WordA will send a pulse of activity through the network. Each word will be activated in turn, and when the last word has been activated, the pulse will stop, and the network will return to its resting state, where no words are ON.

It is fairly straightforward to work out the behavior of this network just by looking at it, and examining how each word impacts on all the other words. With a small network, you can usually do this in your head—or at least with a pencil and

Figure 2. Network 2. A Slightly More Complicated Network.

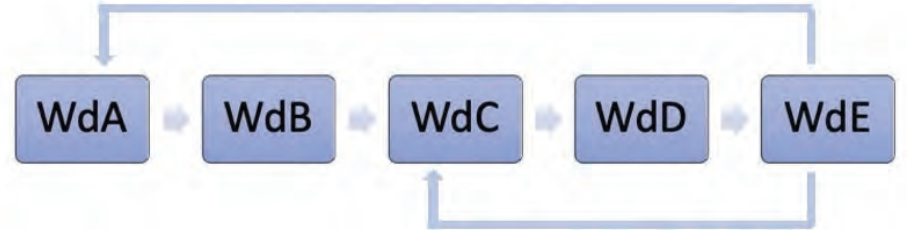

Figure 3. Network 3. Another Slightly More Complicated Network.

paper. However, if we add more connections between the words, then the behavior of the network becomes slightly more complicated. You can see this in Figure 2.

Here we have added just one new connection between WordE and WordA. But this new connection fundamentally changes the way the network behaves. If we turn WordA ON, then the pulse of activation runs through the network as usual, but when WordE gets turned ON, it sends a signal back to WordA and the whole process starts all over again. In this model, the network finds itself in a never-ending loop, where a pulse of activation goes round and round the network forever. Moreover, WordA does not have a special status in this model: turning any of the words ON will start the pulse of activation.

Now look at Figure 3. Before you read on, try and work out how this network behaves if WordA is activated.

In Network 3, activating WordA sends a signal along the network, activating each word in turn. When the signal reaches WordE two outputs are sent out: one goes to WordA, the other goes to WordC. When the network updates itself, both these words are then activated, so we now have two signals in the network, rather than one. The next time the network updates itself, both these signals will propagate through the network. WordA being ON will turn WordB ON, and WordC being on will turn WordD ON. This will move the network into a new state of activity where WordA is OFF, WordB is ON, WordC is OFF, WordD is ON and WordE is OFF. The next time the network updates itself, it will move into a new configuration where WordA is OFF, WordB is OFF, WordC is ON, WordD is OFF and WordE is ON.

Now things start to get much more complicated. You should be able to work out that the network will now move to a configuration where it has three activated Words: WordA will go ON (because it is linked with WordE), and WordD will go ON (because it is linked to WordC). WordC would normally go OFF, but at this

	WdA	WdB	WdC	WdD	WdE
Time 0	0	0	0	0	0
Time 1	1	0	0	0	0
Time 2	0	1	0	0	0
Time 3	0	0	1	0	0
Time 4	0	0	0	1	0
Time 5	0	0	0	0	1
Time 6	1	0	1	0	0
Time 7	0	1	0	1	0
Time 8	0	0	1	0	1
Time 9	1	0	1	1	0
Time 10	0	1	0	1	1
Time 11	1	0	1	0	1
Time 12	1	1	1	1	0
Time 13	0	1	1	1	1
Time 14	1	0	1	1	1
Time 15	1	1	1	1	1
Time 16	1	1	1	1	1

Figure 4. How Network 3 Changes Over Time.

time it is receiving an input from WordE, so it will remain ON. These changes will leave the network in a configuration where WordA is ON, WordB is OFF, WordC is ON, WordD is ON and WordE is OFF. Can you work out what happens next time the network is updated?

You probably found this description confusing and difficult to follow. To be honest, it is really hard to work out in your head how even a very small network behaves. Therefore, it is not surprising that various methods have been devised to show what the network is doing. We have illustrated two of these methods in Figures 4 and 5.

Figure 4 summarizes the behavior of Network 3 when a single word is activated. In this figure, the five words in the network are shown as a horizontal line of 1s and 0s. Each horizontal line summarizes one step in the model. When a word goes ON, it is shown as a red 1; when a word is OFF, it appears as a blue 0. At time 0, there is no activity in the network: all the words are OFF. At time 1, WordA has been turned ON. The remaining lines show how the network responds to this event. This illustration shows very clearly how the network goes from a very low level of activity to a state where all its nodes are ON. Once the network reaches this final state, it stays there forever.

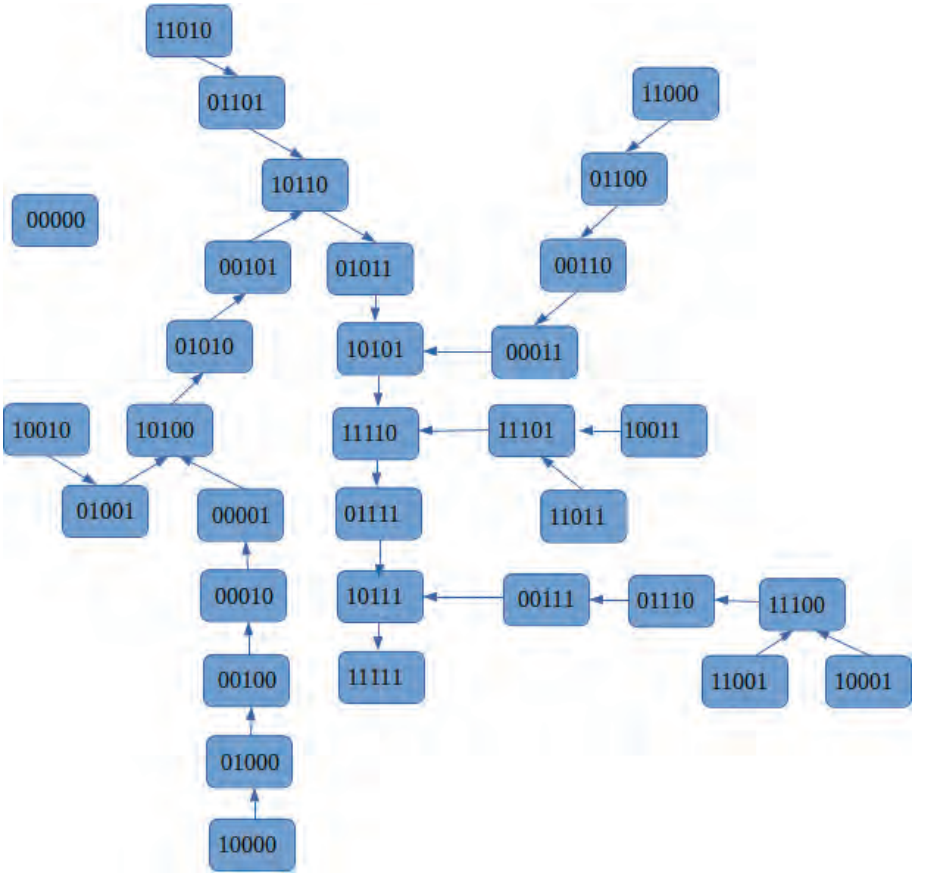

Figure 5. The trajectory map for Network 3.

Figure 5 shows a different way of thinking about the behavior of a network. In this figure, we have a **trajectory map** where the individual cells in Figure 4 have been collapsed into a single row: a string of 1s and 0s. Therefore, the state of the network at time1, with just WordA active, can be described as [**10000**]. At time2 the network moves into a new state where only WordB is active, and this can be described as [**01000**]. At time3, the network is in state [**00100**]. And so on. Each state the network finds itself in has a unique successor state. So, if the network in Figure 3 starts off in state [**00001**], then it will always move itself to state [**10100**].

When you have five words that can take on one of the two values, there are $2^5 = 32$ possible combinations of 0s and 1s. Figure 5 shows each of these 32 states, each linked to its unique successor state, in the form of a trajectory map. It is very easy to see from Figure 5 that there are only two possible long-term outcomes for Network 3. If the network starts out in state [**00000**] then it will do nothing. All the nodes will remain inactive. On the other hand, if it starts out in any other state, then the network will inevitably end up in state [**11111**], where all the words are activated. Sometimes this will happen quickly—for instance, if we put the network into state [**10111**] then it will immediately move to state [**11111**] and

stay there. On the other hand, if we start the network off in state [**10000**] (at the bottom of the Figure) then it will eventually reach state [**11111**] after moving through the 14 intermediate steps shown in Figure 4.

State [**00000**] and state [**11111**] are examples of **attractor states**—states where the network is in a stable configuration, and does not move to a new state. As we will see in the later sections of this workshop, attractor states turn out to be an important concept when we examine the way network models of vocabulary behave.

3.2 Vocabulary Networks

You might now be wondering why we have spent so much time on a tiny model which clearly has nothing to do with L2 vocabularies. The answer is that even this tiny model illustrates some important differences between a heap of words and a network of words. Basically, a heap is inert, whereas a network is usually dynamic, and the behavior of networks can be surprisingly complex. In Network 3 we have only five words and only six links connecting them together, but already it should be obvious that even a network as small as this embodies a huge amount of complexity. And if you tried to work out for yourself how Network 3 responds when it is activated, then you will have realized that it is almost impossible to do the necessary calculations in your head. You can just about manage it when you have only five words in the network, but anything bigger than this rapidly becomes unmanageable.

Real vocabulary networks are obviously a lot bigger than Network 3, and their behavior will be correspondingly more complex. You can get a feel for this by doing some thought experiments in your head. For example, think what would happen to Network 3 if it was in state [**11111**] and one word gets deactivated. Try to work out how Network 3 would behave if you added another connection to the network. Think about how the network would behave if **every** word in the network has two inputs rather than one. Think about how the network would react if some of the links between words are turned words OFF instead of turning them ON. Now try to imagine how a network of 20 words would perform.

At this point, you should be thinking: these small networks are all well and good, but a real vocabulary network is going to contain hundreds of words, not just a handful, and you should be wondering whether size makes a difference. The next sections are designed to let you explore some bigger networks with some extra additional properties. These networks are still a long way off from being plausible models for real vocabularies, but they will prepare you for working with much bigger networks in the later parts of the workshop.

3.3 Some Bigger Networks

We start off by defining some basic properties for these bigger networks:

1: The networks in this section all consist of 100 words.
2: Each word has two inputs from other words in the network.

3: Each word can be ON or OFF.
4: There are two types of words:
Some words go ON if ONLY ONE of their inputs is ON.
Some words go ON only if BOTH of their inputs are ON.

These properties need some explanation:

The ***size of the network*** is basically an arbitrary choice designed to make the visualizations easy to follow and easy to understand. We will look at bigger, more realistic vocabularies in the later sections of this workshop. The question you need to be asking yourself at this point is whether the behaviors we get from a small network will scale up. Would we expect a 1,000-word vocabulary perform in a fundamentally different way from a 100-word vocabulary?

The other three properties define the fundamental characteristics of the words that form the network.

Each of the words in the network have two inputs from other words. This is a step up from the networks that we have looked at so far in this workshop, where most words had only one input. The obvious question to ask here is: why have we limited the number of inputs to two rather than three or four, or more? The answer is that we have done this in the interest of keeping things as simple as possible. We could look at more complex networks where each word has many inputs, and we could look at networks where the number of inputs a word has is allowed to vary—some words might have only one connection, others might have five or six, and a few might have 20 or 30 connections. For the moment, this is a level of complexity that we want to avoid, so we will work with models where each word has just two inputs in the rest of this workshop.

Each of the words in the network can either be ON or OFF. Here too you might be wondering why we are limiting the words to only two activity states in this way. Again, the answer is that we are trying to make our models as simple as possible. We could have invented a more complex scale of activation—for example, we could have eight levels of activation, rather than two—or we could have an activation continuum (language teachers often talk about the active or passive vocabulary continuum). However, if we adopted one of these solutions, then we would have to be explicit about each level of the activity scale or each degree of the continuum. Again, that is a degree of complexity that we will avoid for the moment.

There are two types of words. This third characteristic is a significant change from the network models we have looked at so far, where all the words in the network responded in the same way to their inputs. When words have two inputs, rather than one, we have four different input patterns, and words can respond to these different patterns in a number of different ways. With two inputs (e.g., 0 and 1), we have $2^2=4$ different input patterns (00, 01, 10, 11), and there are 16 (2^4) different ways for any individual word to respond to these inputs. Figure 6 shows the four input patterns and the 16 possible response patterns.

The two columns at the left of Figure 6 show the four possible input patterns. We have two inputs, and each of them can either be OFF (shown as 0) or ON (shown as 1). Each of the 16 columns in the right-hand section of the Figure shows

Inputs		Response Patterns															
		A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**
0	**0**	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	**1**	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	**0**	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	**1**	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Figure 6. The 16 Different Ways that a Word can Respond to Two Inputs in a Network.

0 = OFF, 1 = ON.

one possible set of responses a word could make to these inputs. For example, Column A shows a response pattern where the responding word does nothing: whatever inputs the word receives, it responds by obstinately remaining OFF. In contrast, Column G shows a different pattern of responding. A word of this type goes ON if only one of its inputs is ON, but remains OFF when it receives no input or when both of its inputs are ON. Column I identifies a word that goes ON when both its inputs are OFF, but goes OFF if any of its inputs are ON.

The complexity of these patterns is one reason why the models we will be looking at in this workshop only have two inputs for each word. If we allowed our words to have three inputs, then we would have to think about eight different input patterns, and there would be 32 different ways for a word to respond to these inputs. This is far more than we can handle for the moment. So again, in the interests of simplicity, we will choose to work with just two of the response patterns in Figure 6: Pattern B and Pattern H. Pattern B identifies a word that goes ON only when both of its inputs are activated. Words of this type are conventionally called **AND** words: they go ON if input 1 ***and*** input 2 are ON. Pattern H identifies a word that turns ON if ***any*** of its inputs are ON. Words of this type are usually called **OR** words (they go ON if input 1 ***or*** input 2 ***or*** both inputs are ON). We could use the other response patterns, but for the moment at least, we will stick with just these two. The main reason for this is that it is easy to find a real-world analog for these two response types: Type B words are words that are not easily activated, while type H words are words that activate more easily.

With these basic concepts, we are now in a position to start exploring what a bigger vocabulary network might do. In order to do this, you will need to go to the workshop website: https://www.lognostics.co.uk/Workshop/index.htm

This site contains all the programs that we will be using in this workshop. These programs are designed to run in the **Firefox** browser, and they may not work correctly in other browsers. You will also need a fairly large screen to make them work properly—do not try this on a mobile phone, or a small tablet.

From the Home Page, click on the button labelled **Program-1** ***Basic networks and their properties*** to access the first set of simulations. This program allows you

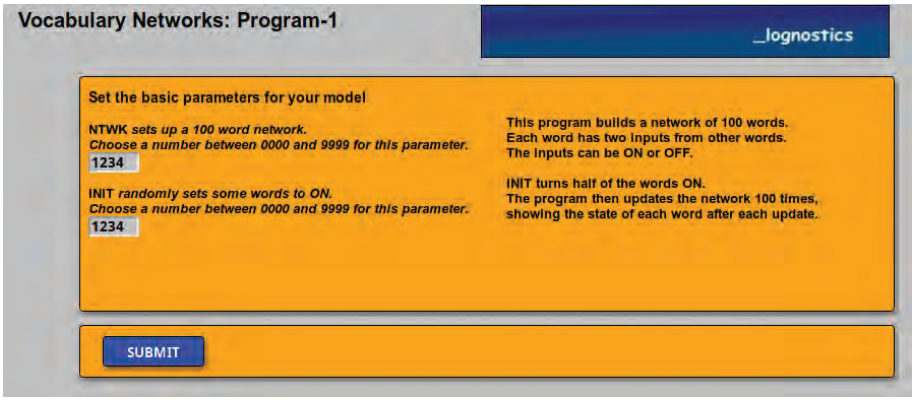

Figure 7. The Start Page for Program-1.

to explore how a large number of networks with the characteristics described above actually perform.

Figure 7 shows the start page for the first simulation set in this workshop. This page lets you vary the values of two parameters, NTWK (network) and INIT (initialization) that define a network. NTWK sets up a 100-word network by deciding how words are connected to other words, and by deciding how each word responds to its inputs. Varying the values of the NTWK parameter will give you a network where the words are connected together in a different way (still with two inputs each), and will also change the way individual words respond to their inputs. INIT sets about half of the words in the network ON. Varying the value of the INIT parameter will set a different selection of words to ON when the program starts up. Leave these parameter values alone for the moment, and just click the SUBMIT button. The program should return an output that looks something like Figure 8.

Figure 8 shows how one instance of a 100-word vocabulary behaves. The top line of the diagram shows the current state of each of the words in the vocabulary after the network has been initialized at random. Dark squares show words that are ON, light squares show words that are OFF. Initially, about 50 words are turned ON. These words send signals to the words they are connected with, as described earlier. The connected words update their activity status in response to these signals, and the network moves to a new state which is shown in line 2 of Figure 8. So, in line 1, Word6 is OFF, but changes status as a result of the inputs it receives from other words, and in line 2, Word6 is ON. Similarly, Word8 is ON at time 1, but is turned OFF at time 2, only to turn back ON again in line 3, which shows the pattern of activation when line 2 is updated. And so on. Figure 8 shows that the number of activated words drops steadily with each successive update. After 9 updates (line 10), only 13 words remain activated. After 14 updates (line 15), only 1 word remains activated. After 15 updates (line 16), the network has reached an attractor state where all the words are OFF and there is no remaining activity in the network.

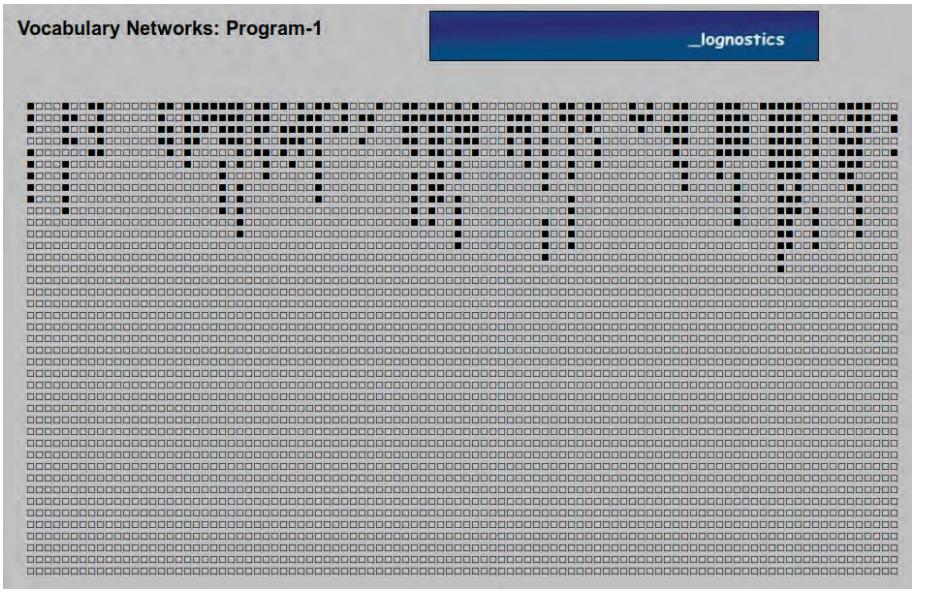

Figure 8. The Output from Program-1 When NTWK: 1234 and INIT: 1234.

However, not all networks behave in this way. You can examine the performance of other networks by changing the value of the NTWK parameter in the Program-1 start page. Changing this parameter gives you a different network: the new network will still contain 100 words, but the words will have different inputs and the way they respond to these inputs will be different. In these first simulations, you do not have any direct control over these characteristics, but each different value of the NTWK parameter will give you a different network.

Figure 9 shows the performance of NTWK 1235. Like the network shown in Figure 8, NTWK 1235 settles into a steady state after a small number of updates, but here, while most of the words in the network have turned themselves OFF, a small number of words seem to be frozen in the ON state.

At this point, you should be asking questions like these:

- *Why do some networks produce words that are permanently ON?*
- *How common are networks that do this?*
- *Do we ever get a network where ALL the items are permanently ON?*
- *What conditions are necessary for this to occur?*
- *Can we move a stable network out of its attractor state?*
- *What conditions would be necessary for a new attractor state to appear?*

You can start to explore ideas like this by playing with the Program-1 and varying the two parameter values on the start page. The NTWK parameter determines how the vocabulary model is set up—how the words are interconnected, and how they react to their inputs. The INIT parameter only changes the initial values that the words in the model are assigned. So, if you run a set of simulations with

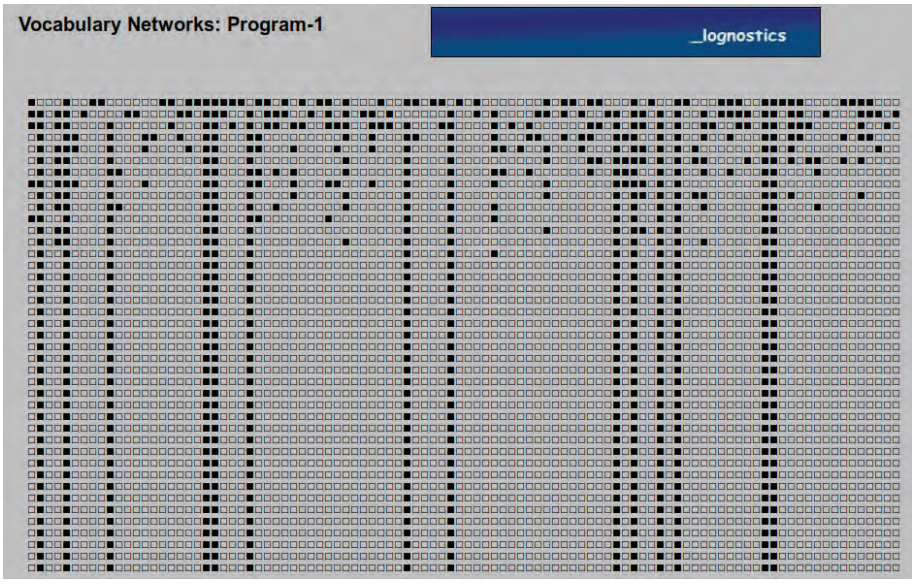

Figure 9. The Output for Program-1 When NTWK: 1235 and INIT: 1234.

NTWK model 1234 you will always get the same model, but you can vary the initial randomization of the words by using different values of the INIT parameter. Similarly, if you fix the value of the INIT parameter, you can vary the value of the NTWK parameter, and see how different models respond to the same initialization. You will probably want to run 20 or 30 simulations in this task to get a feel for how these first models work, and how many different types of performance you can identify. For instance, you could run NTWK 1000 with five or six different values for the INIT parameter (1000, 1001, 1002, 1003, 1004 and so on). Do these different values change the way NTWK 1000 behaves?

There are literally thousands and thousands of different ways to set up a 100-word vocabulary network, even when you have only two characteristics that vary from one network to the next, so it is **very** unlikely that you will find two networks that produce exactly the same results. To help you keep track of interesting models, it is a good idea to start a lab notebook to keep a record of how your simulations are performing. This is not strictly necessary for the simulations using Program-1, but it will be increasingly useful as the workshop models get more complicated. For each simulation that you run, record the parameter settings that you use, and make a note of any interesting features that you notice in the program's report page.

You should find that every different network produces a different outcome. Most of the networks will reach a stable attractor state pretty quickly. All networks with the characteristics described above have at least two attractor states—one where all the words are ON, and one where all the words are OFF. However, most networks end up in an intermediate attractor state where some words are ON, and the rest are OFF. We will look at the implications of this in more detail in part two of the workshop. For the moment, it is worth noting that,

surprisingly, changing which words are ON and which words are OFF when a network is first initialized does generally not result in the network moving into a new attractor state. You can explore this idea for yourself by running Program-1 using different values of the INIT parameter. You should find that changing this value will sometimes result in a network finding a different attractor state, but this does not happen very often. Networks of this size (100 words) typically have only a small number of attractor states. Again, this has some implications for the behavior of real vocabulary networks that we will explore further in Workshop 2.

3.4 Discussion

We began this introduction with a very small vocabulary network which contained only five words. Of course, we are not suggesting that a network as small as this could ever hope to be a good model of a real vocabulary: it is when we start to work with larger networks that some interesting properties begin to emerge that may be relevant to real vocabularies. The most important of these features is that these larger vocabularies are much more stable than we might have expected. Each of the models you worked with in Program-1 were made up of 100 words which could either be ON or OFF, so in theory there are 2^{100} possible states for the network to take up when it is randomly initialized. This is a colossal number of different possible arrangements, and yet after only a handful of network updates a self-reinforcing attractor state always emerges.

Taking all this into account, the suggestion that we want to explore in this workshop is that small network models might be able to act as *vehicles* for an exploration of real vocabularies. Simplified though they are, these models show complex behavior patterns which might throw some interesting light on the way real vocabularies behave.

Let us begin with some simple observations based on the networks we examined in Program-1. These networks have some obvious resonances with real vocabulary networks. It is "obvious" (we will come back to this later!) that real vocabularies consist of a collection of elements that are connected together in some sort of network. It is reasonable to assume that the elements that make up a vocabulary are the words that the owner of the vocabulary "knows." It is also reasonable to assume that these words are "connected" in some way, though we do not know how exactly words in real vocabularies are linked, or how many connections we are dealing with. Clearly, the simple models that we have looked at here do not fully capture this complexity. Indeed, we have made some massive simplifications here: we have not defined what we mean by "a word" or what we mean by "knowing" (recognizing the word, knowing how to use it? cf. Richards, 1976 and Nation, 2001). Nor have we said anything about meaning. But if we suspend our disbelief over these issues for the moment, we can see that simple network models can look quite plausible as analogs of a real L2 vocabulary.

More importantly, accepting this analogy at face value immediately gives us a number of free gifts. The main gift is that it throws some interesting light on the issue of active or passive vocabulary in the mental lexicons of L2 learners, one of

the big theoretical problems in L2 vocabulary acquisition research (cf. for example Melka Teichroew, 1982, 1989 and Melka, 1997).

There is a huge literature dealing with different evaluations of active and passive vocabulary. It is generally agreed that passive vocabulary consists of words that you can recognize but would not normally be able to use without some form of prompting, while active vocabulary consists of words that you can recognize AND use. It is widely taken for granted that a large active vocabulary is a *Good Thing.* It is also widely assumed that all words start off as part of a learner's passive vocabulary and that some of these words eventually end up as active vocabulary items. Furthermore, much research refers to a passive or active vocabulary continuum, where words start off at one end of the continuum and gradually move toward the other end. The question is what drives this change? What makes a passive word permanently active? and how does this change happen? There are no good answers to these questions, partly because the properties of the "continuum" are rarely well-defined in this research, and partly because it is rare to find discussion of any sort of mechanism which would push a word along the continuum. Our model networks begin to suggest a rather different approach to this problem.

To start, it seems reasonable to identify active vocabulary in a real vocabulary network with words that are ON when a vocabulary network is in its stable attractor state. These are words that can be used even in the absence of an external stimulus. We can also identify passive vocabulary as words in a network that are OFF, but can become active when some sort of external stimulus is applied to the network. For example, we could activate an inactive word directly, or we could activate other words in the network that have the effect of causing a passive word to become active (for a while). This is essentially what happens in real life when learners read words: they can often recognize words that appear in a text even if they cannot use them without this supportive context. Note that we do not have to build the idea of active and passive vocabulary into our model as a fundamental assumption: this key idea just appears as an emergent property of a vocabulary network where words have two possible activation states.

We could, of course, design a model vocabulary which has more than two activation states—but it is traditional in simulation studies to keep things as simple as we possibly can until we are forced to make them more complex. In any case, we are interested in the idea that what looks like complex behavior in a vocabulary might actually be generated by interactions between simple components, and it is useful to keep these components as simple as we possibly can. Therefore, for the moment we will stick with a binary passive or active vocabulary.

In fact, the simulations suggest that some vocabulary networks might contain more than two types of word. In some vocabulary networks, we find words which are unstable, sometimes ON and sometimes OFF, sometimes activated, sometimes deactivated in successive updates of their network. This situation arises when a network's attractor state is actually a short cycle of states rather than a single steady state. You almost certainly found some examples of this when you worked with Program-1 on the website. These words, flickering in and out of the active state, seem to make up a significant proportion of any learner's

vocabulary, though the existence of intermittently active words is rarely discussed in the research literature. Our model vocabularies do not just draw our attention to the appearance of this phenomenon: they also provide a natural explanation for why it occurs, and suggest some limitations on the number of intermittently active words we would expect to find in a vocabulary.

And maybe this prompted you to ask:

Why do some words oscillate between the ON and the OFF state?

What conditions are necessary for oscillating words to appear?

How big is a typical set of oscillating words?

In a growing vocabulary, do all words start off oscillating between ON and OFF?

How could we examine the appearance of oscillating words in real life?

What changes would you need to make to a network to make an oscillating word stable?

Do we get words that oscillate together?

Do we get words that oscillate in complementary pairs?

What other structures are likely to emerge in a vocabulary network?

This set of questions is a good example of the way working with models suggests further research, but you should note that some of these questions are very different from the questions that vocabulary researchers have traditionally asked.

Furthermore, the model networks hint that there might be a fourth kind of word. Some OFF words seem to become active as a result of activation by other words in the network (low frequency synonyms, perhaps?), while other words have a very low likelihood of being activated in this way. This sort of situation is likely to arise if the network includes words whose current state depends on a chain of other words which are usually OFF. The chances of one of these words being activated will normally be vanishingly small, especially if the network itself is large, but they will quickly become activated if other words that they depend on are activated. This idea might lead you to ask questions like:

How many other words are typically activated when an arbitrary word is turned ON?

Do networks typically contain words that cannot be turned ON by turning other words ON?

What conditions would be necessary for this to be the case?

What would be necessary to turn these words into active vocabulary items?

What other type of chain effects would you expect to find in a network?

Considerations like these suggest that our simple two state model is actually much richer than it appears at first sight. Not only does it give us a very simple explanation for why some vocabulary is active while other words remain passive, it also suggests that words might naturally fall into other types as well—types which do not arise naturally in the active or passive continuum approach. In this way, the model vocabulary networks highlight a significant gap in our normal thinking about L2 words. More importantly, they provide a relatively simple and straightforward mechanism as to why these different types of words appear in real life: different word types are just an emergent feature of the vocabulary network that the words belong to. The models strongly suggest that it might be wrong for us to think about active or passive status as an intrinsic property of a word. Rather the current status of any individual word is actually a property of the network as a whole, and not a fundamental property of the word itself: if we change the structure of the network by altering the way a word is connected to other words in the network, or by changing the way a word responds to these inputs, then the current activity state of the word is also likely to change, and the activity status of other words is likely to change with it (cf. de Saussure, 1916 for an early version of this idea).

In addition, these simple models suggest that apparently obvious questions about active and passive vocabulary might not be as straightforward as they appear to be at first sight. A simple binary distinction between active and passive vocabulary is already richer than we might have anticipated, and might prompt us to ask whether we really need to have a complex theoretical construct such as the active or passive continuum. This leads on to other questions:

Does it make sense to ask whether the development of active and passive vocabulary is "the same or different"?

Is vocabulary acquisition a single process that just has two different outcomes?

Does it make sense to ask about "the relationship between active and passive vocabulary"? Why would we assume that there is any relationship of this sort?

Does it make sense to ask whether a learner's active vocabulary is always X% smaller- than their passive vocabulary?

On the other hand, the simple models lead us to ask some rather different questions:

How active is an ideal vocabulary?

Is a network where all the words are active "better" or "worse" than one where the level of activity is lower?

What criteria would you use to decide this question?

Is a potentially active vocabulary—a vocabulary which can quickly switch itself into an active state—an efficient solution to the problem of vocabulary storage?

Questions of this sort arise naturally when you run simulations, but they are not questions that can easily be addressed when your main mode of research is large

scale experimental studies, difficult to organize, and often fraught with ethical issues. Working with models allows you to start thinking about the implications of these questions without the huge investment of time and resources that accompany real-life experimental studies.

A second gift that arises naturally in a model vocabulary network is that we find chains of words which are dependent on each other—sets of words which become active only if a particular source word is activated. We will examine this idea in more detail in a later section too. For a moment, let us just note that this might provide a way for words in a vocabulary to naturally form themselves into semantic or thematic clusters—sets of words which are activated together in response to a common source word.

The interesting thing here is that working with even a grossly simplified model vocabulary is already making us think about real vocabularies in a new way. This is going to be a recurrent theme throughout this workshop, and we hope that working through the simulations will fundamentally change the way that you think about vocabularies. For the moment, though, it is enough to note that a relatively superficial consideration of some very simple models of the way vocabularies work has already led us to consider the role of some of the basic constructs in L2 vocabulary research. This is fairly typical of what happens when you work with simulations, and one of the main strengths of this type of work.

Finally, an important idea that runs through this workshop is that working with simulations may appear at first sight to be an easier option than running standard experimental studies, but actually this type of work is not as easy as it looks at first sight. In traditional research, you often do not need to be very specific about how your theory of vocabulary acquisition actually works—it is sufficient to point the reader in the general direction of your thinking without providing much detail. It is also difficult for critical readers to prove that your theoretical model is at fault. This explains why we get so many contradictory results in the research, and why we find so many weird and wonderful metaphors in the research literature. Simulation research does not let you get away with imprecise thinking of this sort. Unless you are absolutely specific about what your simulation has to do, then it just will not work. For instance, you cannot just say "under X conditions words are learned" without specifying exactly what the conditions are, and exactly what you mean by "learned." Furthermore, when you break down a large concept like "learned" into its component steps, you often find that the order in which instructions in a simulation are implemented can sometimes make a huge difference to the way a model runs. Sometimes this forces you to rethink what you are modeling, or it makes you realize that a feature you thought was marginal to your model is actually a crucial feature instead. Or again, you sometimes find that your model only works within a certain range of parameter values. Always, doing simulations means that you have to be absolutely explicit about the assumptions you are making, and this means that your thinking is cruelly exposed to critics. Generally, this is a good thing, though it can sometimes be a bit scary. Simulation research is not an easy option, but it makes you think in a way that traditional research approaches often do not, and the results are often exhilarating and exciting.

The next set of simulations, Part 2 of this workshop, will begin to examine some of the ways we think about L2 vocabularies, and will give you some first-hand experience of just how unsettling working with simulations can be.

Acknowledgements

The authors would like to thank the editors and the two anonymous reviewers for their comments and feedback.

References

Aitchison, J. (2012). *Words in the mind: An introduction to the mental lexicon* (4th ed.). Blackwell.

Collins, A.M., & Loftus, E.F. (1975). A spreading-activation theory of semantic processing. *Psychological Review, 82*(6), 407–428. https://doi.org/10.1037/0033-295X.82.6.407

de Saussure, F. (1916). *Cours de linguistique générale.* Payot.

Melka, F.J. (1997). Receptive vs productive aspects of vocabulary. In N. Schmitt & M. McCarthy (Eds.), *Vocabulary: description, acquisition and pedagogy* (pp. 84–102). Cambridge University Press.

Melka Teichroew, F.J. (1982). Receptive versus productive vocabulary: A survey. *Interlanguage Studies Bulletin, 6*(2), 5–33.

Melka Teichroew, F.J. (1989). *Les notions de réception et de production dans le domaine lexicale et sémantique.* Peter Lang.

Nation, P. (2001). *Learning vocabulary in another language.* Cambridge University Press.

Richards, J.C. (1976). The role of vocabulary teaching. *TESOL Quarterly, 10*(1), 77–89. https://doi.org/10.2307/3585941

Segalowitz, S., & Bernstein, D. (1997). Neural networks and neuroscience: what are connectionist simulations good for? In D. Johnson & C. Erneling (Eds.), *The future of the cognitive revolution* (pp. 209–216). Oxford University Press.