**Introduction**

How is it that we can successfully learn to communicate with one another? The typical formulation of this problem is to ask, for a given language L and a given learner P, how is it that P comes to be a fluent speaker of L? That is, what processes and mechanisms enable P to acquire knowledge of L, its syntax and semantics, so that it can participate in the community of L-speakers? The standard cognitive answer to this is that at least some knowledge of L is innate, or that there are certain constraints on the languages L\* that P can learn such that they can learn L with limited input.

However, there is an antecedent question: how is it that there comes to be a language, L, for P to learn? That is, where do languages come from? The answer that one gives in response to one of these questions typically influences the answer that one will give to the other. If you believe that language is innate, then you will probably think that language comes from inside us. If you believe that language is constructed, then you will be forced to account for the emergence of a language among speakers who do not already posses the language.

**Overview of Phenomena**

For the sake of this paper, I will provide the following simple analysis of a language community: a language community consists of:

* a group of speakers of the language, *P*
* set of meanings, *M*
* set of possible words, *W*
* a set of forward mappings from meanings to words, F1: *M* → *W*
* a set of backward mappings from words to meanings F2: *W* → *M*
* an assignment of forward mappings to the speakers, A1: P → F1
* an assignment of backward mappings to the speakers, A2: P → F2

A language community L is successful if, for any arbitrary pair of speakers, p and q, A1(p) approximates (A2)-1(q) and A1(q) approximates (A2)-1(p), both to some unspecified degree. That is, the forward mapping for all agents are approximately equal to the inverse of the backwards mapping for all other agents in the community.

This raises the question: how should agents go about picking their mappings? Or, for a given dyad, what is it that determines whether a given word, w, is an appropriate label for a giving meaning, m? What criteria should an agent use when selecting a word to communicate a meaning to their partner? In some contexts, there exist fairly reasonable criteria: for instance, when using pictures to communicate the identity of objects, the best picture is that which most closely matches the referred object. When using gestures, often the best gesture is to point at the object. However, in language there is rarely such correspondence between the message and the meaning. Setting aside cases of onomatopoeia or phonetic metonymy (using “meow” to refer to a cat, for instance), most words bare no resemblance to their referent. That is, the word’s relationship to it’s referent is arbitrary.

Given this, how is it that an agent should go about selecting a word, w, to communicate its intended meaning, m? The only rational criterion is whether or not their interlocutor also believes m refers to w. However, the agent’s interlocutor faces a similar problem when selecting a possible referent for w: the meaning that corresponds to wjust is the meaning that the first agent intends to pick out with w. Even granting an agent access to its partner’s mapping function does not help, since neither agent *has* a mapping function to copy.

David Lewis (1969) calls such a problem a *coordination game*, and provides an analysis of such problems from the perspective of decision theory. Briefly, a coordination game is any game that is represented by a pay-off matrix whose off-diagonal outcomes are all zero: that is, the only rational move for an agent to make is the one that their partner makes. The classic example is deciding who should call back when the call is dropped. If both people decided that the person who originally placed the call calls back, or that the person who received the call calls back, then they will be able to get back on the phone. Otherwise, they will either try to call each other simultaneously, and thus shut each other out, or neither will initiate the call. From the perspective of decision theory, there is no strategy that each agent can deploy to maximize their pay-off.

**Prior Literature**

Two prominent views have been advanced in response to this problem. For the sake of space and clarity, I will only summarize two representative positions.

*Lewis (1969): Convention Model*

Lewis (1969), in addition to providing an analysis of coordination games, offered up a tentative solution in the form of *conventions*. Briefly, a convention is a special type of regular behavior. That is, a pattern of regular behavior *R* is a convention in some population *P* if and only if (Lewis 1969):

* everyone in P conforms to *R*
* everyone expects everyone else to conform to *R*
* everyone has the same preferred outcomes
* everyone prefers to conform to *R* on the condition that everyone else in *P* conforms to *R*

A limitation of Lewis’s view is that it is steady state one: that is, it describes a situation in which a coordination game has already been solved, rather than providing a strategy that enables a population to solve a coordination game. However, Lewis model makes strong claims on the kinds of populations that can solve coordination games, in that the agents in the population must be capable of explicitly representing the behavior of other agents in the population. That is, they must have some capacity to represent *common knowledge*, a set of facts *K* such that everyone in the population *P* knows, and that everyone knows that all other members of *P* know *K*.

*Barr (2004): Agent Based Model*

Barr (2004) presented the results of a simple agent based model of convention formation. In his model, interacting agents learned to associate a set of four possible meanings with a set of four possible words. The agents learned associative mappings from the set of meanings to the set of words by interacting with each other and adjusting their associative map to more closely resemble their partner’s. At each time step, each agent selected one partner to communicate a message to, and then updated its mapping based on the outcome of that interaction.

The results from this model were impressive: for populations of 100 interacting agents, the agents on average converged on a common language within 100 time steps. Furthermore, this rapid convergence was robust to changes in the population size: increasing the population to 1000 agents only increased the average time to convergence to 178 time steps.

Barr (2004) compared the performance of his model to one, the Stay-Switch model, which more closely approximated Lewis (1969)’s proposal: if an agent’s mapping fails at any point, the agent switches to another possible mapping based on that mapping’s past successes. The size of the agent’s memory, as represented by how many time-steps of conversations it stores, was varied across simulations. These simulations revealed an intriguing effect: convergence under this learning regime was most robust when the agent’s memory was small. Indeed, convergence for populations larger than 500 failed for any memory capacity larger than 2, and even for smaller populations, the convergence time was roughly quadratic with respect to memory size.

Barr (2004) interpreted these results as providing evidence that convention could emerge from the interactions between pairs of agents with access only to local, and not global, information about the language. Indeed, the fact that convergence in the Stay-Switch model was most robust for small memory sizes potentially indicates that global information is disruptive. This makes sense: if the population as a whole is disordered, then information about the population can only inform you about that disorder.

*Criticisms of Barr (2004)*

However, there are reasons to doubt the generalizability of Barr (2004)’s results, and indeed whether the model that he offers is truly one in which a language emerges from the dyadic interaction of simple associative agents.

First, the language model that Barr used was exceedingly simple: four perfectly distinguishable meanings mapped onto four perfectly distinguishable words. Additionally, either the meanings nor the messages had any internal structure. It is as if the agents in the model only wanted to talk about single objects — CAT, DOG, AVOCADO, etc. Generally, communicative agents want to communicate messages with some kind of structure: CAT AND DOG, THE DOG ATE THE AVOCADO, etc. When messages have this kind of structure, meanings are no longer perfectly distinguishable: CAT AND DOG is very similar to CAT AND AVOCADO, for instance. Furthermore, these meanings are generally mapped to messages with some internal structure and with strong resemblances to other possible messages, as well. Taken together, these details make the problem of learning a mapping more subtle.

Secondly, the form of the associative map used by Barr (2004), and its learning algorithm, imposes a lot of structure on the possible mappings and the time course of the mappings. Briefly, for any pair of agents, p and q, the result of p and q’s mappings from a given meaning are points two points, xp and xq, in R4. The effect of the learning algorithm is to move that those points closer to the mean of xp and xq. At the population level, this guarantees convergence, since after any update, the new positions of xp and xq are at least as far from the mean position of the total population’s mapping for that message as they were before, and quite likely moved closer. This explains the rapid and smooth convergence of the populations in Barr (2004)’s model, since the meaning vector and the learning algorithm implicitly encode information about the population as a whole.

**Present Model**

The goal of the present model was to address the limitations of found in Barr (2004). Specifically, it was designed to examine the formation of linguistic convention under a less constrained meaning-message mapping. To this end, the present model uses an artificial neural network (ANN) to model the mapping from meaning to message. ANNs are simple, yet flexible systems that can learn a range of functions mapping inputs to outputs.

A secondary goal was to examine the formation of linguistic convention when the messages to be communicated have a structure that more closely approximates natural language. In this model, I focused on thought’s capacity for composition: that is, our ability to combine thoughts in a structured way to obtain new thoughts (Fodor 1975). To this end, I created a more complex representation of meanings, which I refer to here as the model’s *c-language*. In addition, the set of possible messages, which I refer to here as the model’s *m-language*, was made more extensive.

*C-Language Specification*

The c-language consists of a set of semantic primitives and a set of syntactic frames that combine these primitives to generate valid expressions in the language. The c-language is exceedingly simple. There are only two types of semantic primitives, objects and relations. A language is fixed by two parameters, N and M, which give the number of objects and relations in the language, respectively. The elements of the c-language, objects and relations, are denoted as follows:

* Objects: Oi, i ∈ { x | x ≤ N}
* Relations: Rj, j ∈ { x | x ≤ M}

In the c-language, there are only 3 possible syntactic frames:

* Oi, i ∈ { x | x ≤ N}
* Oi Oj, i≠j ∈ { x | x ≤ N}
* Oi Oj Rk i≠j, k ∈ { x | x ≤ N}

Thus, in a language with N objects and M relations, the total number of possible meanings is N + N\*(N-1) + R\*N\*(N-1). Furthermore, given the way that meanings are mapped to messages (see below), if two messages contain the same object or relation, then the representation of those messages in the agent’s brain are very similar. Thus, the c-language can be said to be compositional (Fodor 1975). Note that the only difference between objects and relations is their distributional properties: whereas objects can appear in any syntactic frame, relations can only appear in frames with two objects.

*M-language*

The model’s m-language consisted of a number of possible words, the number of which was set by a parameter available to the user. Sentences in the m-language were represented as binary strings; that is, as sequences encoding the presence or absence of a word in the message. Thus, for a language with S words, there are 2S  possible sentences in that language.

*Associative Map*

After all of that, the mapping between sentences in the c-language and m-language is remarkably simple. Each agent was given a pair of multi-layer artificial neural networks, one which mapped sentences of the c-language to sentences of the m-language, and one which performed the reverse mapping. Each node in the input or output layer of the networks corresponded to an element of the c-language or m-language, and were activated when its element was present in a given meaning or message. The network then fed this activation forward according to the weights among the nodes of the network.

One consequence of this representational system is that meanings, and messages, are no longer perfectly discriminable like they were in Barr (2004). For instance, in a language with 4 objects and one relation, the messages {O1 O2}, {O1 O3}, and {O2 O3 R1} would be encoded as {11000}, {01100}, and {10101}, respectively. The binary strings associated with these meanings share some features, which might make it harder for the associate map to take them to distinct messages. The same story holds for messages in the m-language.

Learning was accomplished via standard back-propagation. However, only the meaning to message mapping ever directly received input about other agent’s behavior: after a given interaction, an agent’s meaning to message mapping was updated so that it more closely mapped the meaning to the message that its partner would have generated for that meaning, while it’s message to meaning mapping was updated so that it more closely mapped the message that it would generate for that meaning to the target meaning. Thus, the agent’s implemented a form of ‘self-talk’, where their comprehension networks were update to more closely resemble their production networks.

To examine the role of memory, agents were given a limited store of past interactions that they could practice over. That is, each agent stored a list of results of the previous X interactions, and iterated over this when updating its associative maps. The size of an agent’s memory was varied, and the effect on convergence was examined.

*Program*

The above model was implemented in NetLogo using the LevelSpace extension (Wilensky 1999, Hjorth, Head, Wilensky 2015). The parent model specified the structure of the language, provided a way for agents to communicate with each other, and also gave them the meanings that they sent to each other. Agents were implemented as child models, each with the mapping network described above.

The behavior of the model is exceedingly simple: patches in the world encode possible meanings in the c-language. At each time-step, agents randomly move around the world. They examine the meaning encoded on the patch that they end up standing on. Then, they generate a message that they believe encodes that meaning and sends that message to another agent. That second agent then decodes the message according to its own associative map and checks whether the decoded meaning matches that found in the original patch. Either way, the first agent then updates its associative mappings to better approximate its partner’s.

*Validation*

The first step was to check if a single agent could learn a pre-specified language. Results indicated that agents could not learn a simple pre-specified language. Extensive testing could not identify the results of this failure, but a few possibilities present themselves. One possibility is programmer error, though extended safaris failed to uncover anything glaring. Furthermore, when the agent is given a single meaning to message pairing to learn, it succeeds quite well. Finally, the occasional success of the population model seems to indicate that something about a single agent learning a pre-specified language with random structure leads the learning algorithm to get stuck. Thus, the results of the validation were interpreted as a tentative success.

**Results**

After all of that build up, you’re (hopefully) probably excited to read about the range of cool results that I obtained. Well, I’m sorry to disappoint you: across a range of parameter settings, the population in the model fails to converge at the same rate as Barr (2004)’s. That is not to say that the agents fail to communicate: the average behavior of agents in the models was to successfully communicate at rate above chance, and this rate of partial success was obtained fairly rapidly, usually within 30 time steps. However, this rate did not increase over time, nor did it routinely get very high.

I did, however, replicate Barr (2004)’s original robustness to changes in population size. Interestingly, larger populations seemed to help, to an extent: when the model was run with 2 agents, it frequently failed to communicate at all. One possibility is that the learning algorithm occasionally gets stuck, and having multiple sources of conflicting information helps jostle it out of this well. A trade off is that total convergence was occasionally observed for these small populations, though this was rare.

I did not notice if there was any effect that including relations in the c-language. Models with relations appeared to performance that was similar to other models with the same number of elements in the c-language, though there was some indication that having relations slightly hurt convergence. If relations do not hurt, this might be due to the fact that the only difference between objects and relations was distributional: just so long as the ANNs underlying the model’s performance were able to learn a mapping, the fact that occasionally one more node was turned on in the input layer was of no consequence. If the observed tendency to hurt performance was real, it might be because meanings in c-languages are, on a whole, more similar to each other. Thus, it becomes harder for the agents to work out a mapping that discriminates them. Further, more extensive testing is needed to investigate these possibilities.

**Discussion**

The results of the above model provide weak evidence that the success of Barr (2004)’s model of convention formation is attributable to the simple structure of its meaning and message languages, or to the nature of the learning algorithm that it used. The present model both complicated the language model and simplified the mapping, and observed that this made convergence slower, weaker, and less likely. Because of the way the model is set up, it is impossible to tell whether making the language more complicated or making the associative map simpler had a larger impact on convergence. Other language specifications, and other methods of implementing associative maps, must be explored in order to better understand this issue.

Bibliography

Barr, D. (2004). Establishing conventional communication systems: Is common knowledge necessary? *Cognitive Science*, 28(6), 937–962. <http://doi.org/10.1016/j.cogsci.2004.07.002>

Fodor, Jerry A. (1975). *The Language of Thought*, Cambridge, Massachusetts: Harvard University Press.

Hjorth, A. Head, B. & Wilensky, U. (2015). LevelSpace NetLogo extension [computer software]. Evanston, IL: Center for Connected Learning and Computer Based Modeling, Northwestern University. <http://ccl.northwestern.edu/rp/levelspace/index.shtml>

Lewis, David, 1969. *Convention*, Cambridge: Harvard University Press.

Wilensky, U. (1999). NetLogo [computer software]. Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University. http://ccl.northwestern.edu/netlogo .