The emergence of monotone quantifiers via iterated learning

Carcassi, F.; Steinert-Threlkeld, S.; Szymanik, J.

Publication date
2019
Document Version
Final published version
Published in
Creativity + cognition + computation

Link to publication

Citation for published version (APA):
Carcassi, F., Steinert-Threlkeld, S., & Szymanik, J. (2019). The emergence of monotone quantifiers via iterated learning. In A. K. Goel, C. M. Seifert, & C. Freksa (Eds.), *Creativity + cognition + computation: 41st Annual Meeting of the Cognitive Science Society (CogSci 2019) : Montreal, Canada, 24-27 July 2019* (Vol. 1, pp. 190-196). Cognitive Science Society. https://cogsci.mindmodeling.org/2019/papers/0054/index.html

The emergence of monotone quantifiers via iterated learning

Fausto Carcassi[1,*], **Shane Steinert-Threlkeld**[2,*], **Jakub Szymanik**[2]

fausto.carcassi@gmail.com; {S.N.M.Steinert-Threlkeld, J.K.Szymanik}@uva.nl

[1] School of Philosophy, Psychology and Language Sciences; University of Edinburgh
[2] Institute for Logic, Language and Computation; Universiteit van Amsterdam
[*] Co-first authors.

Abstract

Natural languages exhibit many *semantic universals*: properties of meaning shared across all languages. In this paper, we develop an explanation of one very prominent semantic universal: that all simple determiners denote monotone quantifiers. While existing work has shown that monotone quantifiers are easier to learn, we provide a complete explanation by considering the emergence of quantifiers from the perspective of cultural evolution. In particular, in an iterated learning paradigm, with neural networks as agents, monotone quantifiers regularly evolve.

Keywords: semantic universals; generalized quantifiers; monotonicity; iterated learning; neural networks

Introduction

While natural languages show great variability, there are features that they all appear to share. Linguists call these features linguistic *universals*. Universals have been found at all levels of linguistic structure: phonological, syntactic, and semantic.[1] Some universals might follow from constraints on what humans are physically capable of doing. For instance, there is no language whose prosody requires the production of ultrasounds. The reasons for other universals are harder to understand, leading to multiple proposed explanations.

One well-supported claim is that at least some universals are to be explained in terms of *learnability*.[2] More precisely, it is easier to learn a language that satisfies the universal than it is to learn a language that does not satisfy the universal, and this difference in the complexity of acquisition causes languages that satisfy universals to spread. In the case of universals of lexical semantics such as the one we focus on below, the learnability explanation says that lexical entries whose meaning satisfies the universal are easier to learn, and therefore more likely to be lexicalized. Complicated meanings can be obtained through complex grammatical constructions and compositional interpretation thereof.

The learnability explanation is an empirical, causal claim about the origins of linguistic universals. One way to support the learnability explanation for a specific universal is to provide a model of learning that is cognitively realistic and on which expressions that satisfy the universal are indeed easier to learn.

Finding an appropriate model of learning can however only partially explain a linguistic universal. Learnability is a fact about individual cognition, while a universal is a feature of a whole language. A second challenge consists in connecting these two levels, showing the effects of learnability on emerging language structure. This is the so-called problem of *linkage*.[3]

Iterated learning (IL) is a method that addresses the problem of linkage. In IL, parents teach children their language, who teach the next generation their language, and so on and so forth. The crucial insight of IL is that learning is not an inert process in cultural evolution, since the languages of a cultural child and its cultural parent are generally slightly different. The changes caused by learning are not random, but rather tend to be guided by the learner's cognitive biases. As a consequence, over time languages adapt better and better to the agents' cognitive biases. Learnability can then affect the frequency of different traits.[4]

Previous work has addressed the learnability challenge by showing that quantifiers, responsive predicates, and color terms that satisfy certain semantic universals are easier to learn for neural networks.[5] In this paper, we address the problem of linkage by building an iterated learning model of the evolution of the semantic structure of quantifiers. In particular, we will use neural networks as our agents and standard gradient descent as the learning method inside the context of iterated learning. The next section briefly reviews the theory of generalized quantification and the universal of *monotonicity*. After that, the following section presents the model of cognition and the iterated learning model, as well as an information-theoretic measure of the *degree of monotonicity* of a quantifier. Experiments with this model and their results are presented in the following section. Results are discussed in the final section, along with possible future directions.

[1]For some examples see, respectively, Hyman (2008), Newmeyer (2008), and Barwise and Cooper (1981).

[2]See, e.g., Steinert-Threlkeld and Szymanik (in press), Piantadosi, Tenenbaum, and Goodman (2013), and Peters and Westerståhl (2006).

[3]The problem of linkage was introduced in Kirby (1999).

[4]See, e.g., Tamariz and Kirby (2016); Culbertson and Kirby (2016); Kirby, Cornish, and Smith (2008) for discussions of the way individual cognition is reflected in language structure through IL and experimental evidence supporting the connection.

[5]See, respectively, Steinert-Threlkeld and Szymanik (in press); Steinert-Threlkeld (in press); Steinert-Threlkeld and Szymanik (2019).

Quantifiers and monotonicity

Determiners are expressions that take a common noun as an argument and return a Noun Phrase. Determiners can be grammatically simple—e.g. *some*, *few*, *most*—or complex—e.g. *fewer than three* or *at most five*.[6] Determiners express generalized quantifiers.[7] (Monadic) Generalized quantifiers are properties of sets of subsets of a domain of discourse. The generalized quantifiers expressed by natural language determiners are of type $\langle 1,1 \rangle$, i.e. properties of exactly two sets. Equivalently, a quantifier of type $\langle 1,1 \rangle$ takes (the characteristic function of) a set A and returns a function from (the characteristic function of) a set B to truth values. A is the *left argument* and B the *right argument* of the quantifier. For instance, the sentence "most As are B" is true if and only if the number of As that are B (cardinality of the intersection of A and B, i.e., $|A \cap B|$) is greater than the number of As that are not Bs (i.e., $|A - B|$), i.e.:

$$[\![most]\!] = \{(A,B) : |A \cap B| > |A \setminus B|\}$$

Various universals have been proposed about which generalized quantifiers are expressed by simple determiners. In the following, we focus on the *monotonicity* universal proposed by Barwise and Cooper (1981). This says that all simple determiners across all languages express monotone quantifiers. A quantifier is monotone iff it is *upward* monotone or *downward* monotone. A quantifier Q is upward monotone [downward monotone] iff for any three sets A, B and B', if $Q(A)(B)$ and $B \subseteq B'$ [$B' \subseteq B$] then $Q(A)(B')$. As an example, consider the upward monotone quantifier $[\![most]\!]$. Assume that the sentence "Most cats sleep" is true and that everything that sleeps is alive, i.e. $[\![sleep]\!] \subseteq [\![alive]\!]$. The monotonicity of $[\![most]\!]$ ensures then that "Most cats are alive" is true.

Monotonicity is an interesting universal because it is easy to imagine non-monotone quantifiers. Examples of non-monotone quantifiers abound among the meanings of complex determiners: "an even/odd number of" or "exactly 2", etc. The commonness of non-monotonicity among complex quantifiers makes the lack of simple non-monotone quantifiers especially puzzling and in need of an explanation. Previous work proposed to explain the universal of monotonicity in terms of the greater learnability of monotone quantifiers.

Steinert-Threlkeld and Szymanik (in press) propose to use neural networks in this context. A neural network is a computational device that can learn to approximate functions by observing tuples of inputs and relevant outputs, and progressively minimizing a suitably defined distance between the true output and the network's own prediction. In the case of a quantifier, the input is a structure where the relevant sets are specified and the output is 1 iff the structure verifies the quantifier and 0 otherwise. In practice, given a structure the neural network outputs a probability that can be interpreted as confidence that the structure verifies the quantifier.

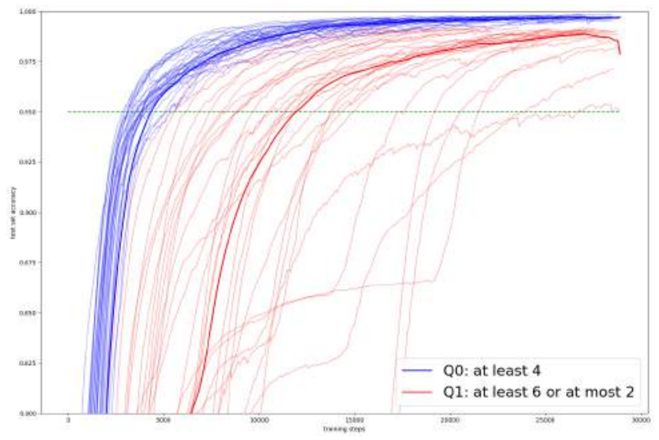

Figure 1: Learning curves on a neural network for the monotone *at least 4* (blue) versus *at least 6 or at most 2* (red). The x-axis is number of training steps; the y-axis is accuracy (percentage correct) on a test set of examples the network has not yet seen. This was Figure 4 in Steinert-Threlkeld and Szymanik (in press).

Data about how fast neural networks learn different kinds of quantifiers was produced with the following algorithms. First, two quantifiers are picked such that one satisfies the universal and the other does not. Then, the two quantifiers are taught to a neural network until it has accurately learned them. The crucial information is how long on average it takes neural networks to accurately learn quantifiers that satisfy the universal compared to ones that do not. Various universals were tested in this way. In the case of monotonicity, the data was produced both for a downward monotone and for an upward monotone quantifier. The neural networks were strikingly faster at learning monotone compared to non-monotone quantifiers. Figure 1 shows an example.

As discussed above, knowing that meanings with certain features can be learned more easily only goes some of the way in explaining the features' universality across various languages. A full explanation also needs to show that the structure can and eventually will be reached by processes of cultural evolution. In the rest of this paper, we develop an iterated learning model of the cultural evolution of quantifiers that embeds the learning model of neural networks, and show that monotonicity reliably emerges.

Methods

Iterated learning

IL models start with two groups of randomly initialized agents, the first and second generations. Each agent in the first generation—the *cultural parent*—is associated with one agent in the second generation—the *cultural child*. A set of

[6]Exactly how to draw the distinction between simple and complex and whether, for instance, *most* is simple or complex, do not matter for present purposes.

[7]For more information on generalized quantifier theory from linguistic, computational, and cognitive perspectives, see also Peters and Westerståhl (2006) and Szymanik (2016).

linguistic production data is generated for each cultural parent and used as input for the cultural child. The cultural child tries to approximate its cultural parent's language. In the following step, the process is repeated with agents in the second generation as cultural parents and the new agents in a third generation as cultural children. The cultural transmission process is iterated for some number of generations. Each cultural family line is called a *chain* of IL.

Crucially, the agents do not learn their parent's language perfectly. There can be various reasons for this. First, there can be a bottleneck in learning. This happens when the learner does not observe everything that is needed to perfectly reconstruct the language, and therefore has to guess some aspects of it. The number of data points given to the learners is fixed for all generations and agents and is called the *bottleneck size*. A second reason is that the agent might not have perfect memory or perfect reasoning abilities, and might therefore learn a language that does not perfectly conform to the given data. In this case, the more rational the agent, the closer the learned language will be to the teacher's language. A third reason is that the cultural parents might produce language in a way that is stochastic rather than deterministic. This can make the language harder to approximate and impossible to learn perfectly. For instance, a cultural parent might pick among the signals compatible with a certain observation according to a categorical distribution. The cultural child would need to infer the parameters of the distribution, a task which cannot in general be accomplished perfectly with a finite number of observations.

The changes introduced by each learner accumulate over generations. Since these changes are not completely random, but rather tend to be consistent across agents, the languages tend to change in the same way over time in different chains. In sum, IL is a way to study how the cognitive system of the learners determine which languages one should expect to see spoken in a population of such agents. The crucial individual level components of an IL model are the set of possible languages, and the way the agents learn them. We now explain these two components in turn.

Model of models, quantifiers, and language

Since the focus is on the evolution of monotonicity, we simplify the language model by assuming that the quantifiers are conservative and extensional.[8] This amounts to saying that the truth value of each quantifier only depends on the elements in A and $A \cap B$, and not on $\overline{A \cup B}$ or $B \setminus A$. Therefore, the truth of any quantifier depends only on which of the elements of A are also elements of B, and which are not. Assuming conservativity and extensionality both reduces the number of possible quantifiers that agents can speak and simplifies the model of each quantifier, since only A and $A \cap B$ need to be encoded. Moreover, we assume that the left argument of the quantifiers is fixed to some set A with cardinality n.

Assuming conservativity/extensionality and a fixed set A, we can represent the part of the world—called a *model*—that is relevant to determining the truth value of a quantifier as a bit vector of a fixed length n. Each element of the model represents an object in A. Each element has value 1 iff the object corresponding to that bit is also an element of B, and 0 otherwise. For instance, the vector $[0,1,1]$ would model a situation where $A = \{o_1, o_2, o_3\}$ and o_2, $o_3 \in B$. The set of models is the set of all binary strings of length n, i.e. the set of possible relations between a fixed A and any possible B. We call M' a *submodel* of a model M iff M' is 0 everywhere where M is 0. For instance, $[0,1,1,0,0]$ is a submodel of $[0,1,1,1,1]$. Note that each model is a submodel of itself.

We represent a *quantifier* as a function from models to single bits. An example of a quantifier is $Q(x) = 1$ if $\sum_{i=1}^{n} x_i > 2$ otherwise 0, meaning "more than two". Since for A of size n there are 2^n different models, each quantifier is a 2^n-sized bit vector. Each element of the quantifier vector corresponds to a model and has value 1 iff the model verifies the quantifier and 0 otherwise.

To see how this works in practice, consider a set A of size 3. There are 8 possible ways in which any other set B can overlap with A. Each of these is modelled as a bit vector of size 3. For instance, $[0,1,1]$ says that the second and third object of A are also elements of B, but the first is not. The English expression "all As are B" is modelled by a bit vector of size 8 that has value 1 at the index corresponding to the model $[1,1,1]$ and 0 otherwise. If the models are ordered lexicographically[9] and the last model is therefore $[1,1,1]$, then the quantifier corresponds to the vector $[0,0,0,0,0,0,0,1]$. We call a quantifier *degenerate* if and only if it corresponds to a vector of identical elements, 0s or 1s. A degenerate quantifier corresponds intuitively to a quantifier that is true or false of every model.

Each agent encodes a single quantifier. Agents do not encode the quantifiers directly. Rather, given a model they produce a truth value by using a neural network. The next two sections clarify the connection between the neural networks and the agent's behaviour.

Neural Networks

Because of the aforementioned learnability results of Steinert-Threlkeld and Szymanik (in press), the agents that make up the generations in our iterated learning setup are *neural networks*. Each network has n input neurons (one for each bit of a vector corresponding to a model) and one output neuron (how probable it thinks that the true output is a 1), with two hidden layers of 16 neurons each. We made this

[8]These, next to monotonicity, are two prominent semantic universals distinguishing natural language quantifiers from all logically possible quantifiers. Extensionality means that extending or shrinking the universe of discourse has no effect on the truth-value of the quantifier sentence as long as the left and right arguments are unchanged. Conservativity means that only the part of B that is common to A matters for the truth-value of the sentences. In other words, the elements in $B \setminus A$ can be safely ignored when determining the truth-value. See Peters and Westerståhl (2006) for definitions.

[9]In that case, lexicographic order is the dictionary order over sequences of letters from the alphabet $\{0,1)$ with 0 preceding 1 in the order.

choice so that the networks had enough expressive power to represent many quantifiers, including complex ones. Future work will analyze the effect of architecture choices on the results presented below. The networks and learning, which will be described in the next section, were implemented in PyTorch.[10]

Such a network learns from input/output pairs using a fancier version of gradient descent called Adam (Kingma & Ba, 2015). The network receives a number of true input/output pairs, which it iterates over in small batches. For each batch, it guesses the correct outputs for the inputs, and then updates its parameters (weights and biases connecting the neurons) in such a way that its future outputs are guaranteed to be closer to the truth.[11] Because this style of learning is fairly gradual, we introduce one more parameter to our simulations, namely *number of epochs*: this is how many times the network processes its training set in each generation. In other words, the network sees a portion of its parent's language, as determined by *bottleneck size*, but gets to learn from that portion number-of-epochs times.[12]

Model of the agents

Each agent plays two roles in an IL simulation. The first role is to learn a language given data from the previous generation. The second role is to produce data used to teach to the following generation. To produce this data, the agent is prompted with randomly chosen models.

In the learning phase, each agent receives learning data consisting of a set of tuples $\langle$model, judgment$\rangle$. The judgment is a single bit expressing whether the quantifier used by the agent is compatible or not with the model. This data is used to train the agent's neural network as described in the previous subsection.

Production works as follows. The agent feeds an observed model to its neural network. The neural network returns a number in the $[0,1]$ interval. Then, the agent rounds the number and returns it. The returned number expresses whether the agent's quantifier is compatible or not with the model that the agent observed. The production behaviour is deterministic, since an agent always produces the same bit given the same model.

Prompted with a string of 1s and 0s, agents produce a 1 or 0. The former models a state of the world, the latter models the compatibility of the agent's quantifier with the world state. However, nothing in the simulation implies that neural networks are interpreting 1 and 0 as True and False respectively in their input and output. Therefore, the output of an agent under-determines which quantifier the agent speaks, even when the output for all models is known. For instance, an agent that returns 1 for input $[0,0,1,1]$ can be interpreted as accepting the model where $B = \{o_3, o_4\}$ (if 1 is interpreted as True in the model and in the quantifier), as rejecting the model where $B = \{o_3, o_4\}$ (if 1 is interpreted as False in the quantifier and True in the models), as accepting the model where $B = \{o_1, o_2\}$ (if 1 is interpreted as True in the quantifier and False in the models), or as rejecting the model where $B = \{o_1, o_2\}$ (if 1 is interpreted as False in the quantifier and the models). Crucially, the interpretation of the bits has to be consistent across the models and across the quantifier judgments. Therefore, each agent can be interpreted as speaking four quantifiers, depending on whether 1 and 0 are interpreted as meaning true or false in the models and in the agent's output. We discuss below how we deal with underdeterminacy when it might make a difference to the interpretation of the results.

Measures of monotonicity

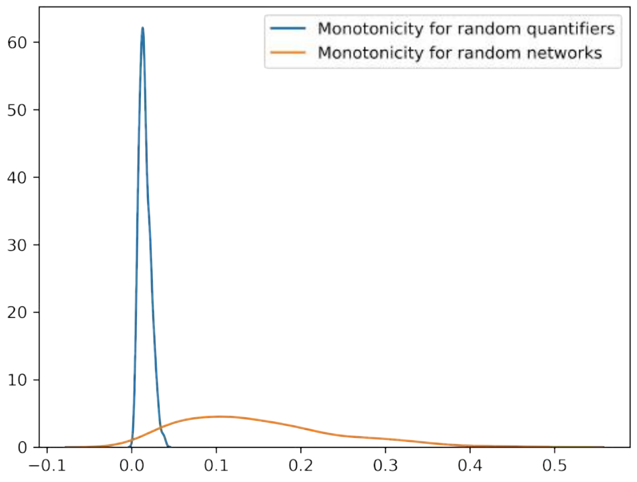

Figure 2: Kernel Density Estimation of the distribution of degrees of monotonicity from a sample of 300 completely random quantifiers and 300 random neural network agents. The x-axis is the measure of monotonicity we describe in the main text.

According to the standard definition, monotonicity is a binary property. A possible way of analyzing the results would be to find the proportion of monotone languages in every generation. However, some quantifiers are intuitively more monotone than other quantifiers. For instance, consider the three quantifiers "some", "between 3 and 5" and "an even number of". While "some" is monotone and the other two quantifiers are not, intuitively "an even number of" is the least monotone of the three. To track finer changes in monotonicity level over time, we define a graded measure of monotonicity.

We measure monotonicity in information-theoretic terms as the proportion of uncertainty in the output of a quantifier that is removed after knowing that there is a submodel where the quantifier is true (i.e. a 1). For a perfectly (upward) monotone quantifier Q, if a model M has a submodel to which the quantifier assigns 1 then Q will assign 1 to M. Therefore, for

[10]http://pytorch.org

[11]For general introductions, see Nielsen (2015); Goodfellow, Bengio, and Courville (2016).

[12]In some experimental literature — for example, Carr, Smith, Culbertson, and Kirby (2019) — this is also referred to as *exposures*.

a monotone quantifier all the uncertainty is removed and the measure has value 1.

More formally, first define the random variables $\mathbb{1}_Q$ and $\mathbb{1}_Q^{\prec}$ on the space of possible models as follows. $\mathbb{1}_Q$ is the value that Q assigns to the model M. $\mathbb{1}_Q^{\prec}$ is whether a model has a submodel that the quantifier considers true (assigns 1 to). The entropy of $\mathbb{1}_Q$, $H(\mathbb{1}_Q)$, quantifies the uncertainty about what truth value Q will assign to a model. The conditional entropy $H(\mathbb{1}_Q \mid \mathbb{1}_Q^{\prec})$ quantifies the uncertainty about what Q will assign to a model, given that one knows whether the model has a submodel that Q considers true (assigns 1 to). $H(\mathbb{1}_Q \mid \mathbb{1}_Q^{\prec})$ is minimized (attains value 0) for a perfectly monotone quantifier: if you know that a model has a true submodel, and the quantifier is upward monotone, you know the truth value of that model. The difference between the entropy and the conditional entropy between these variables is known as the mutual information:

$$I(\mathbb{1}_Q;\mathbb{1}_Q^{\prec}) := H(\mathbb{1}_Q) - H(\mathbb{1}_Q|\mathbb{1}_Q^{\prec})$$

This measures how much information $\mathbb{1}_Q^{\prec}$ provides about $\mathbb{1}_Q$. For a perfectly monotone quantifier, $H(\mathbb{1}_Q|\mathbb{1}_Q^{\prec}) = 0$, and so $I(\mathbb{1}_Q;\mathbb{1}_Q^{\prec}) = H(\mathbb{1}_Q)$. In other words: for a monotone quantifier, knowing which models have a true sub-model provides as much information as knowing the entire quantifier.

While this roughly captures what we want from a measure of monotonicity, it needs to be normalized to form a degree that applies across quantifiers, since $0 \leq I(\mathbb{1}_Q;\mathbb{1}_Q^{\prec}) \leq H(\mathbb{1}_Q)$. We do this by dividing by $H(\mathbb{1}_Q)$, moving the upper bound to 1. In total then, we measure monotonicity as

$$\begin{aligned}\mathsf{mon}(Q) &:= \frac{I(\mathbb{1}_Q;\mathbb{1}_Q^{\prec})}{H(\mathbb{1}_Q)} \\ &= \frac{H(\mathbb{1}_Q) - H(\mathbb{1}_Q|\mathbb{1}_Q^{\prec})}{H(\mathbb{1}_Q)} \\ &= 1 - \frac{H(\mathbb{1}_Q \mid \mathbb{1}_Q^{\prec})}{H(\mathbb{1}_Q)}\end{aligned}$$

To see how this measure tracks intuitions, consider the previous mentioned quantifiers "some", "between 3 and 5" and "an even number of". "Some" gets monotonicity 1.0, because knowing whether a model has a submodel that verifies "some" eliminates all uncertainty about the truth of the model. Recall that each agent can be interpreted as instantiating any of four quantifiers, which can be monotone to different degrees. This raises the question of which of the four degrees of monotonicity should be considered in the analysis of the results. The monotonicity of an agent's language is the highest among the degrees of the quantifiers compatible with the agent's language. For instance, an agent whose quantifier is "between 3 and 5" has degree 0.7517 and one with "an even number of" has degree 0.001.

We compare the results of the simulation to the distribution of the measure in randomly generated quantifiers. There are two different random distributions of quantifiers. On the one hand, there are the quantifiers instantiated by randomly initialized agents. On the other hand, there are the quantifiers sampled uniformly from the space of possible quantifiers. These two distributions are depicted in Figure 2. While the completely random quantifiers have a narrower distribution, both types of random distribution are very skewed towards low degree of monotonicity. This makes sense: monotonicity is a relatively rare property, and so should not be expected to appear randomly. We now turn to the results, showing that higher degrees do emerge via iterated learning.

Materials

For our experiments, we used a fixed model size of 10 (which, recall, is also the size of the input to the agents), with 10 agents in each generation, and varied the bottleneck size (200, 512, 715, 1024) and number of epochs (4 and 8). For each setting of those two parameters, we ran 20 trials.

The code, data, and instructions for running experiments may be found at `https://github.com/thelogicalgrammar/NeuralNetIteratedQuantifiers`.

Results

The first result is that monotone quantifiers evolve consistently and rapidly for some values of the simulation parameters. More specifically, the evolution of monotonicity depends on the bottleneck size and the number of epochs, i.e. how much of the parent's language is observed by the cultural child. See Figure 3 for the results. If the networks get too much input, they learn the quantifier accurately and change is very slow. If the networks get too little input, the learning has little effect and no pattern emerges. If languages are somewhat stable across generations, but enough variation is allowed by not over-training the cultural children, monotonicity evolves.

A second result is that the monotone quantifiers that emerge are in large part non degenerate. In Bayesian models that include a prior for simplicity, degenerate languages become widespread under pure IL (Kirby, Tamariz, Cornish, & Smith, 2015). Here, however, degenerate quantifiers are a small minority (about 0.005% of all quantifiers).

The third result is that most non-degenerate monotone quantifiers fall in one of a few types. About 79% of the perfectly monotone quantifiers show the following pattern: there is some index i such that the quantifier—call it Q_i—assigns 1 to a model iff the model is 1 at i (or an equivalent pattern obtained by switching 0 and 1 uniformly in the models and/or in the quantifier). Q_i is true iff o_i, the object represented by index i, belongs to the set B.[13]. Therefore $Q_i(A)$ functions like a proper noun for o_i. Just like "Anna is human" is true iff Anna belongs to the set of humans, "$Q_i(A)$ is B" is true iff o_i belongs to the set B.

[13]In set-theoretic terms, Q_i is a *principal ultrafilter* If U is a finite non-empty set, a set F is a principal ultrafilter on U if there is an $a \in U$ such that $F = \{B \in \mathcal{P}(U) | a \in B\}$. In the present model, Q_i is (the characteristic function of) a principal ultrafilter on B because it it contains every subset of B that contains i.

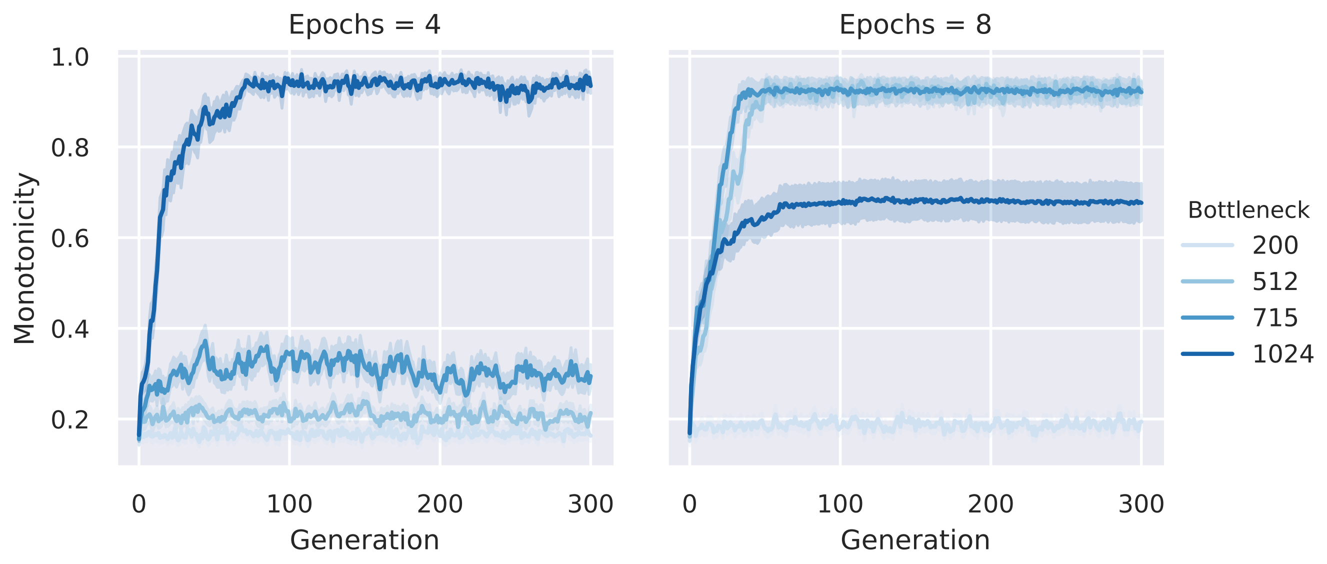

Figure 3: The simulation was ran 20 times for each combination of bottleneck size and number of epochs in a population of 10 agents and a maximum model size of 10. The plot shows how the average monotonicity level across all languages changes over 300 generations. Convergence to monotonicity depends on how much the learners' neural networks are trained, which itself depends on the number of epochs and the bottleneck size. With small bottleneck and few epochs, monotonicity does not evolve. With a bigger bottleneck size and more training epochs, monotone languages become widespread. However, increasing the training data further tends to impede the development of monotone languages.

For other monotone quantifiers $Q_{\{j,k\}}$, there are two indices j,k (with $j \neq k$) such that $Q_{\{j,k\}}$ assigns 1 to a model iff the model has value 1 at both j and k (or, again, an equivalent patterns obtained by switching 0 and 1 in the models and/or in the quantifier). $Q_{\{j,k\}}$ is true iff B contains two specific elements of A, and false otherwise.[14] It can be interpreted as the conjunction of two proper nouns. Like "Anna and Rob are human" is true iff Anna is human and Rob is human, "$Q_{\{j,k\}}(A)$ is B" is true iff o_j is B and o_k is B.

Discussion

The results we presented support the learnability account of the origins of semantic universals of quantification. While previous work compared quantifiers satisfying semantic universals to quantifiers that do not, we have presented a model where the former are selected out of all of the possible quantifiers by a process of cultural evolution. Moreover, the preference for monotone quantifiers is not a consequence of an explicitly coded bias for simplicity, but rather of an independently motivated, biologically plausible model of learning. The results therefore suggest that not only are monotone quantifiers easier to learn, but they are also widespread in language *because* of their learnability.

This model can be straightforwardly extended in various ways. The agents judged their quantifier compatible with a given model simply by rounding the output of their neural network. An alternative to this is for the agents to accept a model with a probability proportional to the network's output. Such so-called sample agents do not straightforwardly instantiate a quantifier, since they can produce inconsistent output when repeatedly prompted with the same model. However, preliminary results have shown that neural networks are capable of doing *statistical learning*: given enough data, they approximate not just whether their parents tend to reject or accept a model, but also the probability of acceptance.

While the quantifiers that emerge from our experiment are monotone, they are unnatural in certain respects. For instance, the proper-name-like quantifiers that emerge are not *quantitative*, i.e. their truth value depends not simply on the number of 1s and 0s, but on the identity of particular elements.[15]

To try and explain the emergence of quantifiers which are both monotone and quantitative, it might be necessary to make it more difficult for the networks to rely on the identity of particular objects by, for instance, shuffling the order of models in the parent and the teacher's inputs. Another pressure that might contribute to shape the meaning of quantifiers comes from communication (Kirby et al., 2015). While

[14]These are called in set-theoretic terms *principal filters*. They are not principal ultrafilters because their truth depends on more than one element.

[15]See Steinert-Threlkeld and Szymanik (in press) for the definition of and motivation for quantity, which generalizes the isomorphism/permutation constraint in generalized quantifier theory as discussed, for instance, in Peters and Westerståhl (2006).

some semantic universals of quantification might have an advantage in cultural evolution because they conform well with learning biases, other universals might evolve because they lead to more successful communication. Therefore, combining iterated learning with a pressure for accurate communication might help more natural quantifiers emerge. We leave all of these exciting possibilities to future work.

Acknowledgments

Shane Steinert-Threlkeld and Jakub Szymanik have received funding from the European Research Council under the European Unions Seventh Framework Programme (FP/20072013)/ERC Grant Agreement n. STG 716230 CoSaQ.

References

Barwise, J., & Cooper, R. (1981). Generalized Quantifiers and Natural Language. *Linguistics and Philosophy*, *4*(2), 159–219.

Carr, J. W., Smith, K., Culbertson, J., & Kirby, S. (2019). Simplicity and informativeness in semantic category systems. Retrieved from https://psyarxiv.com/jkfyx

Culbertson, J., & Kirby, S. (2016). Simplicity and Specificity in Language: Domain-General Biases Have Domain-Specific Effects. *Frontiers in Psychology*, *6*. doi: 10.3389/fpsyg.2015.01964

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. The MIT Press. Retrieved from https://www.deeplearningbook.org/

Hyman, L. M. (2008). Universals in phonology. *The Linguistic Review*, *25*(1-2), 83–137.

Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *International Conference of Learning Representations (ICLR).* Retrieved from https://arxiv.org/abs/1412.6980

Kirby, S. (1999). *Function, Selection, and Innateness: The Emergence of Language Universals*. Oxford; New York: OUP Oxford.

Kirby, S., Cornish, H., & Smith, K. (2008). Cumulative cultural evolution in the laboratory: An experimental approach to the origins of structure in human language. *Proceedings of the National Academy of Sciences*, *105*(31), 10681–10686.

Kirby, S., Tamariz, M., Cornish, H., & Smith, K. (2015). Compression and communication in the cultural evolution of linguistic structure. *Cognition*, *141*, 87 - 102. doi: https://doi.org/10.1016/j.cognition.2015.03.016

Newmeyer, F. J. (2008). Universals in syntax. *The Linguistic Review*, *25*(1-2), 35–82. doi: 10.1515/TLIR.2008.002

Nielsen, M. A. (2015). *Neural Networks and Deep Learning*. Determination Press. Retrieved from http://neuralnetworksanddeeplearning.com/

Peters, S., & Westerståhl, D. (2006). *Quantifiers in Language and Logic*. Oxford: Clarendon Press.

Piantadosi, S. T., Tenenbaum, J. B., & Goodman, N. D. (2013). Modeling the acquisition of quantifier semantics: a case study in function word learnability. Retrieved from http://colala.berkeley.edu/papers/piantadosi2012modeling.pdf

Steinert-Threlkeld, S. (in press). An Explanation of the Veridical Uniformity Universal. *Journal of Semantics*. Retrieved from https://semanticsarchive.net/Archive/DI5ZTNmN/UniversalResponsiveVerbs.pdf

Steinert-Threlkeld, S., & Szymanik, J. (2019). *Ease of Learning Explains Semantic Universals.* Retrieved from https://semanticsarchive.net/Archive/zM5ZGIxM/EaseLearning.pdf

Steinert-Threlkeld, S., & Szymanik, J. (in press). Learnability and Semantic Universals. *Semantics & Pragmatics*. Retrieved from http://semanticsarchive.net/Archive/mQ2Y2Y2Z/LearnabilitySemanticUniversals.pdf

Szymanik, J. (2016). *Quantifiers and Cognition. Logical and Computational Perspectives*. Springer.

Tamariz, M., & Kirby, S. (2016). The cultural evolution of language. *Current Opinion in Psychology*, *8*, 37-43. doi: 10.1016/j.copsyc.2015.09.003