Iterated Learning Model: A Review

Submitted to **Dr. Roni Katzir** by **Tomer Filiba**

Abstract

In this report, I survey Simon Kirby's, Henry Brighton's and their colleague's approach to the Iterated Learning Model (ILM), as presented in a series of papers from 1999 to 2005. Of particular interest are the two papers, *Learning, Bottlenecks and the Evolution of Recursive Syntax (2002)* [Kirby 2002] and *Language as an Evolutionary System (2005)* [Brighton 2005]. Many of their other joint papers (see the references at the end of this review) represent different stages in the research of the University of Edinburgh group, and I chose to focus on the two extremities. It should be noted that ILM has many other contributors, like Batali (who used Elman-style neural networks) and Niyogi (who investigated populations of speakers and learners), but this paper does not cover their work directly.

ILM itself is a very "intuitive" model, in the sense that it seems to mimic the way languages are acquired in humans, passing from one generation to another, changing in the process. It offers some interesting perspectives on how languages (might) evolve, what constrains the structure (syntax) of languages, and how recursive and compositional structures emerge on their own, given some reasonable constraints of the environment.

I will also offer my criticism to Kirby's and Brighton's models, which stems from the way agents *learn* and some of their presuppositions. I argue that their results are of great value and have excellent explanatory power, but they answer different questions.

Introduction to Iterated Learning

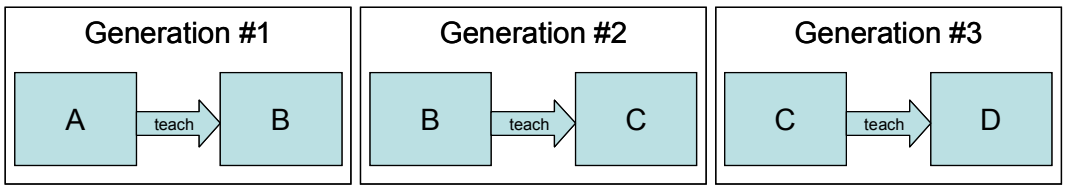
ILM is grounded in the works of Pinker and Bloom [Pinker Bloom 90] and Hurford [Hurford 90], who suggested analyzing languages as the result of a culturally-transmitted, **evolutional process**. Much like biological evolution, languages can be viewed as the result of cultural-selection over many generations of speakers and learners: unlike most animal languages, which are taken to be innate, human languages must be **re-acquired** by each individual.

This leads to the conclusion that languages must **adapt** to fit various constraints that govern their cross-generational transmission: they must be **learnable** by language-learners and must allow effective communication between proficient language-users (*producers*).

ILM builds on the notion that languages have two distinct levels of representation [Chomsky 86, Hurford 87]: an abstract, internal level of *meanings* (also *logical form* or *(I)nternal Language*), and a concrete level of *utterances* or *signals* (also *(E)xternal Language*). *Meaning* can never be transmitted directly – it cannot be serialized in its pure form; the only way to convey meaning between two entities is by producing suitable signal(s).

The simplest form of ILM consists of a "proficient" *agent* that produces pairs of signal and meaning, which are used to "train" a new, clean-slate agent. The new agent is exposed only to this sample data and is expected to have the capacity of learning the

speaker's language. In the next iteration, the proficient agent is removed and the trained agent takes its place, being now "responsible" for training yet another new agent. We call each such iteration a *generation*, and this model is exemplified in the diagram below:



In this form, there is only one proficient speaker and one learning agent in each generation. The model can of course be enriched to support a population of proficient speakers, so that learners are exposed to variation in their parents' language (and perhaps even their grandparents' language), but such models (although viable) are more complicated and Kirby's team has usually preferred to use the simplest form.

For our purposes, the simple model is enough to investigate the properties achieved through ILM, namely, it is "complex enough" to give rise to compositional and even recursive syntax.

Language as a Mapping

Languages can be viewed as a mapping between meaning and signal: an agent, when prompted with a meaning, is expected to produce a corresponding signal. The properties of this mapping are of interest on their own: for instance, if the mapping is many-to-one, multiple meanings produce the same signal, making the language ambiguous and more difficult to learn [Brighton 2005].

The **meaning (*semantic*) space** is inherently different in the two papers (and will be discussed separately), but for the time being, we only need to assume it has some internal structure. The **signal space**, however, is ubiquitously taken to be that of non-empty strings drawn from some alphabet, $\sum$. Generally, we will favor shorter strings, to better emulate the words of natural languages.

Using this simple setup, we can already identify two types of languages:

- **Holistic languages** are much like dictionaries, mapping meanings to arbitrary strings. For instance, the meaning of "John likes Sue" could be mapped to "apcj" while the meaning of "John sees Sue" could be mapped to "kq" – there is no correlation whatsoever between the structure of meaning and the produced signal [Kirby 2002].
- **Compositional languages**, on the other hand, offer a strong correlation between the "parts" of the meaning and the parts of the signal. For instance, if "John likes Sue" maps to "bxt" and "John likes Mary" maps to "gxt", then we can expect "xt" to mean "John likes", where "b" refers to Sue and "g" refers to Mary (at least in this context). Here, the meaning of the whole signal is determined from the meaning of its parts and the way they combine with one another. In fact, this is exactly how Kirby defines compositional languages [Kirby 2002].

An interesting observation can be drawn by inducing a "metric" on the semantic and signal spaces: a mapping that correlates "distance relationships" is more likely to produce a compositional language. In other words, a language would be compositional when the mapping produces a strong correlation between the distance of meanings and the distance of signals [Brighton 2005]. Simply put, if "small changes" in the meaning

translate to "small changes" in the signal, it is more likely that each "part" of the meaning maps to a "part" of the signal. The opposite is also true: when the mapping weakly-correlates distances between the meaning and signal spaces, we're more likely to end up with a holistic language. We will get back to this observation later on.

Learning, Bottlenecks and Evolution (2002)

In his 2002 paper, Kirby outlines an interesting yet simple setup that is capable of developing the hallmark properties of human language: compositionality and recursive grammar.

Semantic Space

The semantic space of choice in this paper is that of zeroth-order predicate logic (predicate logic minus quantification). This space consists of atoms, representing either individuals or two-place predicates, which combine to form formulas. For example, P(a,b) is a valid formula in this meaning space (given that P is a predicate and a and b are individuals).

The space has the notion of a *degree*, which specifies the maximal number of nested formulas allowed. For instance, P(a,b) is a degree 0 formula, while P(a,Q(b,c)) is of degree 1, and so on. For technical reasons, a predicate's two arguments must always be different (i.e., $a \neq b$). It is also important to note that this space **may be infinite**, due to the recursive construction of formulas (if unbounded).

The Experiment

The experiment follows the setup of the single-parent model outlined previously: an agent produces pairs of meaning and signal, from which a new agent learns, who later becomes the producer. Agents have an internal grammar (a variant of context-free grammar, discussed in more detail in the next section) that is decorated with semantic labels. When a meaning is given to the agent, it attempts to find the suitable production from its grammar rules (according to the semantic label), or produces a random string.

Agent 0's grammar is empty, which means all of its productions would be random strings. However, the learning agent, unlike its predecessor, is exposed to a sample of meanings and signals, from which it develops a primitive form of grammar. One the whole sample is observed, this grammar undergoes the process of **rule subsumption** – a set of heuristics employed to generalize it as much as possible.

A crucial parameter of this process is the **learning bottleneck**, which, in general terms, specifies what proportion of the language-space agents are exposed to. For instance, assuming a degree-0 semantic space of 5 predicates and 10 individuals, the set of "well formed sentences" in the language is $5 \times 10 \times 9 = 450$. The question then becomes, how many sentences is the learner is exposed to, during the learning period? Although intuitively we may assume "the more the merrier", the size of the learning bottleneck has profound implications on the developing languages.

Representation of Grammar

In its most basic form, the grammar is made of a start rule, S, which has multiple productions. All productions are decorated by a semantic label, which associates a meaning with each utterance. For instance:

S/love(john,mary) → ksjds
S/love(bill,mary) → qt

This reads, S can be rewritten as *qt* and the meaning of which is *love(bill,mary)*. If a production consists of non-terminals, they are semantically-labeled as well:

S/love(john,mary) → A/john B/love C/mary
A/john → ezj
B/love → ka
C/mary → iuds

And lastly, we can use variables in the semantic labels, to enable a more general form of grammar:

S/x(y,z) → A/y B/x C/z

This way, when we have a meaning such as *see(bill, joe)*, *see* maps to *x*, *bill* to *y*, and *joe* to *z*, so we will look for the productions *A/bill*, *B/see* and *C/joe*. Basically, the grammar can be viewed as a semantically-extended CFG, which proves useful in this experiment.

Grammar Induction

Rule subsumption, or *grammar induction*, is the process that generalizes holistic rules into more compositional ones. In essence, it looks for similar parts in the structure of the semantic label and the production and attempts to correlate them: when two rules are semantically-different by exactly one atom and there is a single, consecutive string of terminals that sets their productions apart – we assume the two are correlated. Therefore, we extract the two rule's common parts into a new rule, remove the previous rules, and add a new non-terminal to the grammar. This is better outlined by an example. Given the two productions:

S/eats(tiger,sausages) → foobarspam
S/eats(john,sausages) → baconbarspam

We generalize them into

S/eats(x,sausages) → A/x barspam
A/tiger → foo
A/john → bacon

This process is called *chunking* and it is detailed in the appendix of Kirby's paper. Chunking is complemented by *merging* and *simplification*, which unify identical rules and further extract their common parts out, resulting in an even more general grammar.

Invention

When an agent is asked to produce a signal for some meaning, it will try to look it up in its grammar. If there is a "path" in the grammar that constructs the requested meaning, the generated string is returned. However, when no such path exists, the speaker will resort to *inventing* a word for it. The invented word (a random sequence of alphabet symbols) will then be fed into the speaker's grammar and another round of learning will occur. This means the **speaker** will induce more grammar from this invented word, thus enriching its own as well as the learner's language.

Pseudo Code

While this setup may seem complicated, it can easily be expressed in pseudo code. Note that the code below neglects the notion of agents – it only maintains a grammar, which changes in each iteration.

```
function ILM(semantic_space, generations, bottleneck):
    grammar ← ∅
    for i = 1 to generations:
        grammar2 ← ∅
        for j = 1 to bottleneck:
            meaning ← random_meaning(semantic_space)
            signal ← produce(grammar, meaning)
            grammar2 ← grammar2 ∪ (S/meaning → signal)
        grammar ← subsume_rules(grammar2)
    return grammar
```

The grammar begins empty, which means all calls to *produce* would return a random word, but later iterations would work with a partial grammar. When the process ends, we're interested in the resulting grammar (used by the last generation), which contains the fully-developed language.

Results

Before we can discuss the results of Kirby's experiments, we must get back to the **learning bottleneck** (a parameter for the *ILM* function above). This parameter controls the number of meaning and signal pairs that learners are exposed to and from which they induce their grammar. At the two extremes, the results are uninteresting: given a very low bottleneck, learners are exposed to very few examples, from which it may not be possible to generalize. This means that when an agent attempts to produce a signal for a novel meaning, it's most likely it would have to invent one. This results in a highly unstable language that changes considerably from generation to generation, and whose chances of convergence are very low.

On the other hand, if the bottleneck is very wide, there's little pressure to generalize – the agent can simply record the whole language, verbatim. However, this language is expected to be **stable** from generation 1, as the learner is exposed to the vast majority of the possible meaning and signal pairs: later generations would use induce virtually the same language of their parents, as there's little need for invention.

The most interesting results, therefore, arise when the bottleneck is low but wide enough to allow a representative sample of the language to pass through. The exact numbers change considerably in different experiments, and are highly dependent on the size of the semantic space and the heuristics employed by grammar induction. However, when the conditions are right, the properties we sought after emerge on their own: the language becomes compositional and recursive!

We can think of the bottleneck as an evolutionary pressure on languages to adapt and become more regular, which means that a smaller sample can be used to reconstruct (most of) the original grammar. When the bottleneck is wide, such pressure is negligible and languages remain of a holistic nature. When it is too narrow, not enough samples are witnessed to notice recurring patterns. But when enough evidence is given to the grammar inducer, compositional structures emerge and are evolutionary-favored (as all other structures won't fit into the bottleneck). Therefore, the languages that "survive" this process must be reconstructible from a rather small sample size, which results in

compositionality – the meaning of the whole signal is a function of the meanings of the signal's parts.

Recursion is but a higher degree of compositionality: it allows the embedding of compositional structures into one another. When the learner is exposed to meanings such as know(john, love(bill,lucy)) or know(bill, say(john, love(joe, mary))), the most compact grammar it could induce to accommodate such structures must be recursive – any other grammar would simply be larger.

And indeed, after several hundreds/thousands of generations, the resulting grammar shows a high degree of compositionality and recursion, drastically reducing its size and becoming very stable – it can efficiently be transmitted across generations, fitting into the bottleneck.

Language as an Evolutionary System (2005)

In this paper, Henry Brighton (of Kirby's team) further develops his 2003 PhD dissertation and provides a comprehensive review of ILM with respect to language evolution and how different parameters control it.

Semantic Space

The semantic space used in this paper is radically different from the *recursive semantic space* presented in the 2002 one. Here, meanings are seen as vectors in an F-dimensional space (of features), each feature having V possible values, yielding a finite space of size V^F. For example, the vector <1,2,1> could identify a "meaning coordinate" in a space spanned by F=3 and V=2.

Note that we cannot express recursive meanings, like "John knows (Bill loves Sue)" in this meaning space, and that all meanings are of the same "length" or "complexity".

Measuring Compositionality

Given the structure of this space, we can define a metric on it using the *Hamming distance*: the distance between two vectors is the number of changes required to transform one into the other. Similarly, on the signal space, it is more intuitive to use Levenstein distance (edit distance) as the metric: this metric measures minimum number of insert, delete and substitute operations required to transform one string into the other.

Using these two metrics, we can calculate the Pearson correlation between pairs of meanings and signals. That is, we take all pairs of meaning, $\forall i \neq j \ <m_i, m_j>$ and generate their respective signals, $<s_i, s_j>$. Next, we compute the correlation of the two vectors:

$$C = corr(<d_H(m_i, m_j) \mid i \neq j>, <d_L(s_i, s_j) \mid i \neq j>)$$

It so happens that $C \approx 1$ for compositional languages (where a small change in meaning translates to a small change in signal and a bigger change in meaning translates to a bigger change in signal), and $C \approx 0$ for holistic ones (there is no correlation between changes of meaning and changes of signal).

Associative Matrix Model

The first model employed in the paper is that of learning by *associative matrices*: given that languages define a mapping between meanings and signals, we can use matrices to specify the "strength of association" of this mapping. For instance, a_{ij} gives the "strength" of the association of meaning *i* with signal *j*. This simple approach is

frequently used to study the evolution of signal systems, where *production* chooses the "best" signal associated with an input meaning, and *learning* is the process of adjusting the weights of the matrix to accommodate the observed data. In order to allow for structured meanings (vectors, in our case), the model is extended a little to support "wildcards", allowing the use of underspecified features in the meaning vector. For instance, <1,*,2> would match <1,1,2> and <1,2,2>.

The subtle details of this formalism are beyond the scope of this review. However, it's important to note that it is a **parametric model**, which lends itself easily to investigating the properties of the resulting languages. In fact, it is geared towards explaining how learning biases affect linguistic evolution.

For instance, by choosing $\alpha>\delta$ for the parameters of the learning process, we can favor compositional languages over holistic ones, which results in the *ability to generalize*. Other assignments fail to do so. Moreover, by choosing $\delta>\gamma$, we can bias against many-to-one mappings (where multiple meanings map to one signal), favoring unambiguous languages.

These consequences are useful for modeling and investigating learning biases, but we are mostly concerned with their results, as they are quite general and we would expect to find them in most other models as well:

- Learners must have the **capacity to generalize** – without it, compositional structure could never evolve and languages could not stabilize given the learning bottleneck.
- Learners must be biased **against** acquiring **many-to-one** mappings. Not being biased against such mappings would result in a language that's neither compositional nor communicatively functional.
- Learners must be biased **against** acquiring **one-to-many** mappings. If learners were not biased that way, a system of meaning-signal mappings could not be acquired, since it's likely that learners would only be exposed to small subset of the possible signals associated with a meaning.

An ILM model where agents are equipped with these three components, can lead to stable, compositional languages that pass through the learning bottleneck.

It is important to note that natural language **does allow** one-to-many as well as many-to-one mappings. For instance, synonymy (the meaning of *dog* can be realized as *dog*, *hound*, *mutt*, etc.) or the plural/present-tense suffix in English (table***s***, walk***s***), where two meanings map onto one morpheme. However, such examples are usually exceptional and it is well known that languages "hate" perfect synonymy [Markman 89]. Moreover, Brighton argues, the end-state of the language does not allow a perfect insight into the biases at play during the early stages of acquisition – and in fact, it seems children **are** biased against many-to-one and one-to-many mappings. This can explain the historical evidence of languages having the tendency to loose such ambiguous mappings in favor of one-to-one.

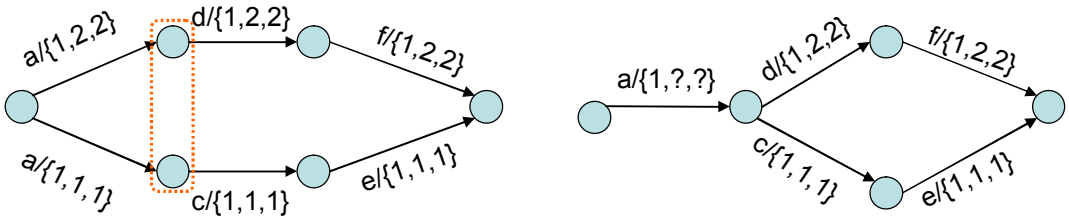
MDL Learning

A different approach to learning, which offers a more "natural setting" than matrices, is that of **Minimum Description Length** (MDL). MDL is a search-space technique that seeks to minimize the combined length of a hypothesis and the encoding of the data by that hypothesis. Formally, given a hypothesis H and data D, the "best hypothesis" in the hypotheses space is the one that satisfies $\text{argmin}_{H \in HSpace}\{L(enc(H)) + L(enc(D|H))\}$.

A very large hypothesis may be able to encode its data succinctly (as holistic languages do), while a very small hypothesis may be too general and require a very long encoding of the data. MDL seeks to find the shortest encoding of the data without hiding too much complexity in the hypothesis itself. In fact, it is easiest to think of MDL as a **compression scheme**, which offers an interesting perspective on learning, whereby generalization equals compression. In other words, a bad hypothesis fails to generalize much, leading to more verbatim (longer) encoding of data; a good hypothesis captures more generalizations, thus better compressing data. The **expressivity** of a general hypothesis is larger than that of less general ones: when faced with unobserved input, the more general hypothesis is more likely to be able to handle it well.

In order to employ the MDL methodology, Brighton introduces a model of Finite State Unification Transducers (FSUT). A FSUT is fundamentally similar to a finite state automaton (FSA), where transitions between states are possible when an input symbol is matched; however, unlike FSA, transducers also produce output during transitions. For instance, suppose you can go from q_1 to q_2 if the input symbol is *a*, and by following that edge you produce an output symbol *b*. Paths must begin in a designated *start state* and end in an *accepting state*.

In our case, the input symbol come from the signal, and the output represents the meaning associates with it. We start by constructing the *prefix tree transducer*, which represents the *maximally specific hypothesis* – that is, it describes the data verbatim as a prefix tree (instead of using a table representation). For instance, considering the language L = {adf/{1,2,2}, ace/{1,1,1}}, we would construct the transducer depicted to the left below:



The *maximally specific hypothesis* is the largest **consistent** hypothesis that describes the given language; in other words, it fully describes the language at hand (and perhaps more, but not in this case). Once we have this, we can start compressing the hypothesis, and by doing so, we are very likely to generalize the language.

We define two consistency-preserving operations: *state merge* and *edge merge* that are used to unify (hence FSUT) recurring parts. For instance, in the previous diagram, the FSUT to the left contains two "equivalent" states (surrounded by the dotted orange rectangle). We can therefore merge the two states and intentionally "under-specify" the meaning associated with *a*. This brings us to the right-hand side diagram, where *a* corresponds to {1, ?, ?}, which leaves the last two features unspecified. The transducer has reduced in size while remaining consistent with the original language.

We can continue with this process, compressing the transducer, as long as it contains equivalent states. It should be noted that this process preserves the consistency of the FSUT with respect to the language, but it may also lead to *generalization*, which happens when we under-specify "too much". This is similar to the *don't-cares* used to simplify Karnaugh maps – when the language does not distinguish a certain feature, we may oversimplify it to achieve better compression. Once we've oversimplified, we allow new paths through the FSUT, which weren't possible before, thus we've extended the range of meanings the FSUT can deal with. If the language data observed during

learning exhibits regularity, it is likely that by applying the aforementioned unification operations, it would end up as a compact, generalized FSUT.

An important difference that sets the MDL learners apart from the association matrices described previously is the ability to **invent**. We've discussed invention in the 2002 paper, but in the two models outlined here, it requires different treatment. First, in the association matrices, there is no inherent difference between invention and induction: the matrix is always "full", so when we present it with a meaning vector, we will always find a suitable "best" signal (note that there could be multiple "best" signals). Whether the matrix has "learned" this association through induction or not, is not apparent in the process.

When a FSUT is asked to produce a signal for an unobserved meaning, and such a path was not formed during unification, we require "true invention". The paper deals with two invention schemes, one that simply generates a random string and registers it with the FSUT, and another that adheres to the MDL principle: it chooses a string such that if it were present in the observed data, would not lead to an increase in the number of states of the FSUT. In essence, it uses the existing structure of the FSUT to generate a new signal; if no such string can be found, the FSUT does not produce anything. The latter scheme proves to converge better.

When taking into account the learning bottleneck of language acquisition, it is easy to see why the MDL fits ILM perfectly: a compressible language (which exhibits higher regularity and internal structure) could easily pass through the bottleneck, as fewer samples can communicate the entire span of the language. Uncompressible, holistic languages cannot "squeeze" themselves into narrow bottlenecks, leading to their gradual decay.

It is also quite trivial why the FSUT approach favors (is *biased* towards) one-to-one mappings. First, one-to-many mappings are simply impossible due to the deterministic nature of FSUTs: they can produce only one signal for a given meaning. As for being biased against many-to-one mappings, these will eventually become under-represented in the sample and simply could not persist: the probability of each generation getting a sample where the same two meanings map to the same signal, quickly approaches zero under uniform distribution.

Comparison of the Two Papers

The two papers suggest radically different implementations of ILM. For instance, the semantic space and induced grammar described in the 2002 paper allows the evolution of recursive syntax and sentences of variable length, while the two models outlined in the current paper allow only fixed-length meanings and sentences. On the other hand, the 2005 paper offers a more parametric treatment to language evolution, which allows the authors to investigate how different biases develop and how compositionality can be measured.

However, the papers share the same "philosophy" of ILM: language is re-learnt generation after generation by "children" from their "forefathers", and the learning bottleneck plays a crucial role in both. Obviously, the 2002 paper can be seen as an earlier, less polished version of the theory (but which has virtues on its own). From a personal correspondence I've had with Dr. Kirby, he explains

> One of the unsatisfying aspects of the model [2002 paper], I now think, is the somewhat arbitrary nature of the heuristics I had to put in place. That's why, later, in work with Henry Brighton we moved on to using MDL to search for grammars instead (although that makes everything a lot slower!).

ILM and Universal Grammar

ILM suggests plausible mechanisms with which a compositional (and even recursive) language could emerge from the "senseless mass" that is holistic languages. In fact, it seems the way languages are forced to adapt for their transmission medium could have given rise to at least some of the language universals we consider innate. In the words of Deacon [Deacon 97]:

> Grammatical universals exist, but I want to suggest that their existence does not imply that they are prefigured in the brain like frozen evolutionary accidents [...] they have emerged spontaneously and independently in each evolving language, in response to universal biases in the selection processes affecting language transmission.

Given that only learnable language could evolve in humans, and that humans have learning biases, it is plausible that not all of the universals of UG ought to be hard-coded in our brains: they could have evolved independently in each language, resulting in the same configuration, much like the evolution of the eye is believed to have occurred independently in many species [Haszprunar 95]. This view is of significant importance, as it reopens the debate of which parts of UG are innate and which are the results of exposure to language itself.

The Learning Bottleneck

We've seen how the learning bottleneck affects the type of languages that evolve in ILM. For instance, it is clear why a narrow bottleneck drives languages towards compositionality – it forces them to "squeeze" into it, thus biasing in favor of regularity and compositionality, as they compress better.

The learning bottleneck closely relates with the **poverty of stimulus**, which is often portrayed as an inexplicable phenomenon that requires innate UG: since there are only so many possible languages, a rather small sample can teach us a lot. However, the way languages evolve under ILM provides a totally different look on the subject: **the poverty of stimulus is a necessity**. Without it, we wouldn't have the drive to develop compositional languages.

If a child were exposed to all of the adult's language (assuming for the time being that it is finite), he or she could simply memorize it all like the agents were doing. Because exposure to language is limited, language had to develop a learnable, compressible structure – in other words, become compositional. Having a compositional language, recursion can be seen as the next logical step – in fact, it may be argued that the unbound nature of human language is itself a **result** of the learning bottleneck.

That is not to say, of course, that children's ability to learn such a magnificent, unbound structure from a finite amount of noisy utterances, should be taken for granted. There is still much to be learned, as far as learning goes.

Criticism

In this section I wish to provide my criticism to Kirby's and Brighton's approach to ILM. While the implications of their research are profound, I argue they provide answers to different questions and make some background assumptions that are not so obvious.

The Compositionality Creep

The semantic space used by both papers is compositional by nature. This is fair considering we normally interpret language using a formal logic system, all of which exhibit compositionality. However, I argue, given a compositional semantic space, compositionality is **bound to pervade syntax**. Kirby and Brighton have built an evolutionary system that favors compactness and regularity (due to the learning bottleneck), so if the semantic side of the mapping exhibits compositionality, any compression scheme applied to the mapping as a whole would surely rely on this structure, leading to structure creeping into the signal space as well. The structures of the semantic space "projects" (or coerces) itself onto the syntax.

It may seem plausible to stipulate that compositionality is "built in" into the semantic system and thus is should be taken for granted. But then again, what surprise do we have in syntax "evolving" to exhibit compositionality or recursion? If the environment "rewards" such structures, such features would surely emerge as the language adapts. Also, if we stipulate recursion into the semantic system, why not stipulate it directly into syntax? Once we have that cognitive capacity, it can be taken from granted.

To further illustrate my point, suppose the semantic space had thematic roles labeled on predicates, e.g., `eat(john:agent, apple:patient)`, so arguments are unordered. It is very reasonable to expect languages evolving over this semantic space to exhibit **case markings**. Does it prove case markings have "evolved" to serve some purpose, or was the system simply biased towards making them explicit? We normally don't mark thematic roles on predicates in our logical systems – we rely on linear order – which leads to linear order in Kirby's experiments (SOV/VSO etc.). Using a case-marking syntax, thematic relations could be alleviated of linear order, so it might seem "reasonable" to do so – but it won't materialize without the necessary evolutionary pressure. In other words, it would not develop in syntax unless the structure is already apparent in the semantic space.

Paired Meaning and Signal

In both models, the learner is exposed to both I-language (semantics) and E-language (signals). Agents basically learn to associate meanings with signals, but for that, obviously, they require observing both sides to the mapping. How can this model ever be extended to a "real-life" learner that is exposed only to E-language? It could be argued that physical objects have an "independent" semantic representation, i.e., the mental object that two people associate with "this apple" is the same (and Kirby does touch that). So, for instance, we can argue that we show "this apple" to the learner and say "apple", which associates the signal with the meaning. Some meanings are simply ground into the "world", and the semantic space can be assumed to simply have them. However, can be extend this formulation to abstract notions, like "yes" and "no"?

The Latin Argument

Again, it could be argued that such abstract concepts are built-in in our semantic/logic system, and that we learn to associate the speaker's intonation with it. To counter this, we can use the *Latin Argument*: it is said that Latin (and several other languages for that

matter) do not have words for "yes" and "no" [12]. Instead, one communicates agreement with sentences like "I concur" or using echo responses ("Is the sky blue?" → "The sky is blue").

First of all, it poses a problem in that the semantic space may change from language to language: some languages would be based on a semantic space where "yes" is a known concept while other won't. More so, how can we explain native speakers of two or more languages? Does it imply that each language has a separate semantic space?

We may try to answer this by saying "yes" exists in every semantic space, but not all atomic concepts are mapped to single words or even phrases. For instance, the meaning of "yes" associates with "I concur" as well as with "the sky is blue" (but also, of course, with "the sea is wet" or "the sun is hot"). Essentially, there is an infinite number of true sentences that "yes" associates with. But then again, a signal such as "the sky is blue" also derives from other meanings, like BLUE(sky). This yields a many-to-many mapping, which should not be learnable at all according to Brighton's analysis.

What is Learning?

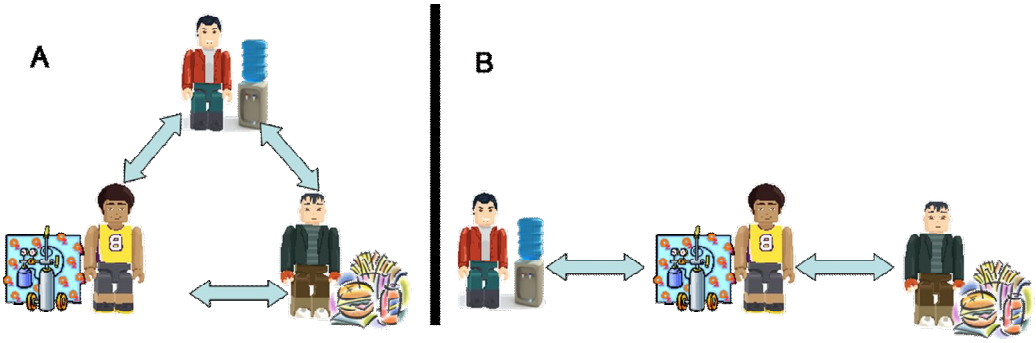
The incident that provoked this review of mine was Kirby's response to my email: "[…] the somewhat **arbitrary nature of the heuristics** I had to put in place". I was trying to replicate his 2002 paper when I realized the paper was lacking many details of how exactly the grammar inducer works, and I contacted Kirby looking for a reference implementation.

It then stuck me that the grammar inducer described in the appendix was really just a collection of ad-hoc heuristics, layered one on top of the other in an attempt produce an over-generalizing grammar. I found it hard to call this process **learning** – it was a carefully crafted mechanism that intentionally over-generalized. Combined with the learning bottleneck, it was always able to find a non-contradictory hypothesis to describe the sample of language given to it. This, along with the structure of the semantic space creeping into the signal space, ultimately summed to the formation of compositional/recursive syntax.

Kirby kindly referred me to more recent research of an MDL ILM learner by Brighton, which he felt was more well-founded. However, I the same happened again: the FSUTs are compressed using over-generalizing consistency-preserving operations, tailored for the task by Brighton. I find it hard calling such schemes *learning*, as I would expect learning to be autonomous, independent of the human who conceived the algorithm. It is Brighton who realized how FSUTs can be compressed, and it is Kirby who came up with chunking and merging of semantically-decorated CFGs. The agents don't learn – they simply follow suit.

An Example of Language Evolution

As an example of what I would consider learning, consider a system of genetically-evolving agents who must negotiate with one another over resources: in order to survive, each agent must poses enough of each resource. Initially (figure A below), agents could be connected in a full-mesh, so any two agents can communicate directly and invent ad-hoc protocols. On the other hand, a "linked-list" topology (figure B) would provide the strong evolutionary pressure towards the formation of a commonly agreed-upon, recursive language: in this case, agents will have to deliver messages **through** one another, resulting in "sentences" of the form "agent 2 says (agent 1 says (give me food))".



This is but a sketch of course, and I have never got to implementing it to see how it evolves or converges. It may very well be the case that this system is too complex for agents to ever evolve a functional language in the first place. However, assuming it did work, we could add new agents to the system, who would first only "listen" in hope of acquiring the language, and only then would they become part of the network, in place of a an existing agent. I suspect different network topologies could affect the properties the languages/protocols that would emerge in the process. For once, they would have to be learnable and enable "tunneling". I would consider this as autonomous learning.

What Questions does ILM Answer

Although I find this model imperfect, there is no doubt ILM provides powerful insights on learning and is able to model the cross-generational cultural-evolution of language and the factors that control it.

For instance, it explains how learning biases affect language acquisition and therefore language in general (as all languages must adapt for successful acquisition). It also introduces the notion of the learning bottleneck, and shows what a powerful drive it has towards more regular, compositional languages. And last but not least (and despite my reservations) it shows how compositional and recursive structures, which are considered to be the hallmarks of natural language, can emerge on their own, given evolutionary pressure.

Summary

The Iterative Learning Model is rooted in the notion that natural language is the result of an evolutionary process, where languages must adapt for successful cross-generational transmission (acquisition). Kirby's and Brighton's work shows how a relatively simple model, made of a single "proficient" agent producing a sample of utterances for a "newborn" agent (which later becomes a proficient speaker on its own), is enough to exhibit the hallmark features of natural language, namely compositional and recursive syntax.

ILM demonstrates how an unstructured, holistic language can develop into a simply-structured, compositional grammar, by assuming a minimal amount of biases and the capacity of agents to generalize. It shows how abstract notions like generalization can be quantifiable using compression: data is compressible when it has an internal structure that can be captured by a hypothesis – so the hypothesis that compresses the most, is the one that generalizes the most. Although an elegant and model, I did find some problems with it which I outlined in the previous above.

References

1. Kirby, Simon 2002. **Learning, Bottleneck and the Evolution of Recursive Syntax**; In Briscoe, T., editor, Linguistic Evolution through Language Acquisition: Formal and Computational Models, chapter 6, pages 173-204. Cambridge University Press.
2. Brighton, Henry, Smith, Kenny, and Kirby, Simon. 2005. **Language as an Evolutionary System**; *Physics of Life Reviews*, 2:177-226.
3. Kirby, Simon, Smith, Kenny, and Brighton, Henry. 2004. **From UG to universals: linguistic adaptation through iterated learning**; Studies in Language, 28(3):587-607.
4. Kirby, Simon 1999. **Learning, bottlenecks and infinity: a working model of the evolution of syntactic communication**; In Dautenhahn, K. and Nehaniv, C., editors, Proceedings of the AISB'99 Symposium on Imitation in Animals and Artifacts.
5. Markman, Ellen 1989. **Categorizing and Naming in Children: Problems of Induction**; MIT Press
6. Deacon, Terrence 1997. **The symbolic species**; Penguin, London
7. Chomsky, Noam 1986. **Knowledge of Language**; Praeger
8. Hurford, James 1987. **Language and Number: the Emergence of a Cognitive System**. Cambridge University.
9. Hurford James. **Nativist and functional explanations in language acquisition**; Logical issues in language acquisition, Rocal M, editor, Dordrecht:Foris
10. Pinker S, Bloom P 1990. **Natural language and natural selection**; Behavioral and Brain Sciences 13(4):707–84
11. Haszprunar 1995. **The mollusca: Coelomate turbellarians or mesenchymate annelids?**; In Taylor, Origin and evolutionary radiation of the Mollusca. Oxford University Press (from Wikipedia)
12. http://en.wikipedia.org/wiki/Yes_and_no#Related_words_in_other_languages_and_translation_problems