Toward the Emergence of Quantifiers

Wesley Rose

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2021

Reading Committee:

Shane Steinert-Threlkeld, Chair

Emmanuel Chemla

Program Authorized to Offer Degree:
Linguistics

University of Washington

Abstract

Toward the Emergence of Quantifiers

Wesley Rose

Chair of the Supervisory Committee:
Shane Steinert-Threlkeld
Linguistics

This thesis explores factors influencing the emergence of quantifiers in a signaling game. It includes a few novel contributions. First is a new signaling game, called the Quantifier Game, designed to provide a setting where quantifier emergence is one natural possibility. It also introduces a new metric designed to measure how well a certain attribute has emerged in a given language, which is applied to measure the emergence of quantifiers. A third contribution is the introduction of a novel loss designed to encourage shorter messages in emergent communication systems. I explore the effects of message compression, curriculum learning, and iterated learning on quantifier emergence. I find that quantifiers do not convincingly emerge in any experiments, but that select implementations of iterated learning do have a small but significant positive effect on quantifier score.

A secondary result is that I find no positive effect of iterated learning on topographic similarity, an effect that seemingly contradicts many recent results showing such a positive effect. I suggest as an explanation that iterated learning's effect on languages could actually be to minimize the number of tokens per relevant feature, something that also results in increased compositionality in some but not all cases.

TABLE OF CONTENTS

LIST OF TABLES

ACKNOWLEDGMENTS

I'd like to thank all of the people that helped me get here. Shane Steinert-Threlkeld for his amazing guidance, introducing me to Emergent Communication and helping me craft a coherent thesis out of a confusion of ideas. Emmanuel Chemla, for providing thoughtful feedback and discussion on early drafts. Thanks to my cohort in the CLMS program for the stimulating discussions and companionship through so many late night coding sessions. Finally, thanks to my family and to Nicole Steinle for encouraging me to go back to school and supporting me through the whole process.

Chapter 1

INTRODUCTION

The evolution of language has been described by Christiansen and Kirby [6] as "The hardest problem in science." This is because it touches so many different disciplines and involves speculation about aspects of ancient history for which there is no historical record.

While the problem hasn't gotten easier, recent advances in deep learning have given language evolution researchers a new tool: Emergent Communication [26, 19]. Emergent Communication refers to a field that explores how *artificial agents* (e.g. neural networks) will learn to interactively develop a communication system, often generously called a language, in different settings. This overlaps with research on the evolution of language because it gives researchers a way to test how certain hypothetical pressures on early humans affect actual emergent communication systems. This is of course a far cry from directly studying precursors to human language, and is certainly not the only tool necessary for this research, but it gives us a way to answer the questions of whether certain pressures theoretically *could* have encouraged human language to evolve in certain ways.

A typical approach to studying language evolution in this way is to first pick one important feature of human language and define it as precisely as possible. Once that feature has been defined for human language, think about how it might appear in more primitive communication systems, such as those that neural network agents would develop. Then think about why this feature of human language might have come about as a part of our evolutionary history. Once a researcher has a theory about the effect of a certain pressure on a certain feature of language, they can design controlled experiments using computational models to see whether implementations of that pressure have the hypothesized effect on emergent languages. More complex experiments can study more than one pressure, asking

the question of whether it is the interaction of multiple factors that encourages a certain feature to emerge.

The most popular feature of human language to study in this way is compositionality. Szabó [28] summarizes the principle of compositionality as the idea that the meaning of a complex expression is determined by its structure and the meaning of its parts. Researchers in language evolution have been theorizing about compositionality for quite some time, and in recent years there has been an explosion of emergent communication research focused on compositionality. Less popular in recent work is studying *non-trivial* compositionality in particular, which describes composition of meanings in a non-intersective way [27]. One way to study non-trivial compositionality is to identify specific elements of language that exemplify non-trivial compositionality, and look for pressures that encourage the emergence of these precise elements.

This is exactly what my thesis does with quantifiers, a case study in non-trivial compositionality. I aim to explore the question of which pressures on human language evolution could have encouraged quantifiers to emerge, and study some of these pressures in a carefully designed emergent communication experiment. Specifically, I will explore whether compression of the communication channel, curriculum learning, and iterated learning have any impact on the emergence of quantifiers.

In the next section, I will present a literature review to situate this work within the body of existing research. Then I will explain the methodology of my experiments. This section will introduce a new signaling game, and describe the variables that will be adjusted between experiments. Following this will be the results and a discussion in which the results will be interpreted. Finally, I will conclude by moving back to the big picture, summarizing my findings and looking forward to how future work can expand on some interesting themes.

Chapter 2

LITERATURE REVIEW

One of the most common approaches in emergent communication research is to use signaling games or referential games, as originally conceived by Lewis [20], and studied in depth by Skyrms [26]. The term signaling game describes a framework in which there are at least two players. The first player, called the sender, has in mind an action that it wants the other player, the receiver, to perform. The only way it can communicate with the receiver is with a signal, or a message. The receiver then takes an action, and both players are rewarded if the action is correct. Recently, a particularly active line of research has been using this framework to study compositionality. Out of this recent research, there are a few themes that are quite relevant to this thesis and which will be canvassed in turn: The distinction between different types of compositionality, methods for concretely measuring compositionality, and adjustable factors that have been shown to affect the structure of emergent communication systems, including the compositionality of these systems.

2.1 Types of Compositionality

Steinert-Threlkeld [27] describes two distinct types of compositionality: trivial and non-trivial. He states that a language is trivially compositional "just in case complex expressions are always interpreted by intersection (generalized conjunction) of the meanings of the parts of the expression." Another way to think about trivial compositionality is that each part of a message contributes a meaning towards the whole independently of each of the other parts. Chaabouni et al. [5] describe this type of compositionality as "sets or sequences of primitive elements" where each symbol reliably refers to a primitive attribute of the object space. A prototypical example of trivial compositionality that is often studied in emergent

communication is the combination of a color attribute and a shape attribute, such as *red circle*. This expression can be understood as the intersection of the set of *red* objects and the set of *circular* objects.

Non-trivial compositionality, on the other hand, "involves non-conjunctive modification of one linguistic item by another" [27]. In a system of non-trivial composition, there must be primitive elements in the messaging space that do not refer to trivial attributes in the meaning space. Instead, these message elements modify primitive attributes in a meaningful way. As an example, consider *fake boat*. This expression can't be understood as the intersection of *fake* objects and *boats*. Instead, *fake* is modifying *boat* in a more complex way. This distinction between *trivial* and *non-trivial* composition will be an important theme throughout this thesis.

2.2 Methods for Measuring Compositionality

Whether one is interested in trivial or non-trivial compositionality, an enduring challenge is how to measure this feature of an emergent language. As mentioned in the introduction, Szabó [28] summarizes the principle of compositionality as the idea that the meaning of a complex expression is determined by its structure and the meaning of its parts. As Chaabouni et al. [5] point out, this definition assumes full knowledge of the primitive expressions and their combination rules. In other words, in order to determine whether a complex expression is compositional, a researcher would need to know the lexical meaning of every primitive symbol as well as the syntax used to combine them. Furthermore, the definition from Szabó is binary; a language is either compositional or not. It would be more useful to be able to define compositionality on a scale.

As a solution to this, Brighton & Kirby [3] present the idea of topographic mappings for measuring compositionality. If language is thought of as a mapping between a signal space and a meaning space, they describe a compositional language as a language where "similar signals map to similar meanings". The definition is deliberately non-specific, and relies on each practitioner to precisely define the meaning space and the distance measures, but given

these specifics one can then take any two signals and any two meanings and measure how similar the two distances are, thus arriving at a quantification of compositionality. This was an important development because it is a concrete metric that exists on a scale. In the literature, this metric is alternatively called topographic similarity, topological similarity, or simply `topsim`. It has started to be used as a standard, but it does not always match intuitive judgements. For example, Chaabouni et al. [5] describe two languages, one that transparently concatenates symbols in a fixed order and one that mixes deletion and insertion operations on free-ordered symbols, which would have the same topographic similarity but intuitively seem to exhibit quite different types of compositionality.

In an attempt to find a metric that more closely matches intuitive judgements, Chaabouni et al. define two alternative compositionality measurements which are inspired by disentanglement in representation learning: positional disentanglement (`posdis`), and bag-of-symbols disentanglement (`bosdis`). `Posdis` measures "whether symbols in specific positions tend to univocally refer to the values of a specific attribute" by measuring the mutual information between each attribute in the object space and each position in the message space. `Bosdis` similarly uses mutual information but treats order as irrelevant, instead focusing on symbols as they occur anywhere in a message. These disentanglement metrics are useful because, much like `topsim`, they allow a quantitative comparison between languages, and they intuitively seem to measure something closer to the compositionality that researchers are interested in. However, as Chaabouni et al. themselves acknowledge, these metrics are only designed to measure trivial compositionality.

Measuring non-trivial compositionality or overall compositionality is more difficult because the metric needs to capture the mapping of messages to meanings that are not primitive attributes of the input. A second shortcoming of this metric is that it can be maximized without information about all attributes being communicated in the messages. As an example, consider positional disentanglement in an object space with two attributes. For each position, the metric finds the absolute value of the difference in mutual information between the two attributes, or $|MI(pos, att1) - MI(pos, att2)|$, normalized by entropy. It then av-

erages that score across all positions. If every position conveys maximal information about attribute 1 and no information about attribute 2, the posdis score will be 1, the maximum. However, this system would not seem compositional because primitive meanings are never actually composed. The messages only convey the same primitive attribute meaning with every position. Although looking at accuracy can help to detect systems like this (a receiver who gets no information about attribute 2 will have trouble picking the correct object in a signaling game), it remains a potential problem with the metric.

One idea for capturing non-trivial or more general compositionality is to measure generalizability of languages, or more specifically the ability of agents that develop a language as an aid to performing a task to generalize to performing the task for novel inputs. Researchers commonly cite compositionality as the reason that humans are able to produce and understand novel inputs (e.g. Lake et al. [17], Cogswell et al. [7]), so this idea makes some intuitive sense. However, Chaabouni et al. cast doubt on this idea. They design an experiment using a sender and a receiver agent who are learning to reproduce inputs. They carefully withhold certain inputs during the training phase, and then measure generalizability by testing the agents' ability to successfully recreate the withheld inputs. Concurrently, they measure the compositionality of the language developed by the agents using `posdis`, `bosdis`, and `topsim`. They find that compositionality measured this way does not predict generalizability, which is an important result suggesting that computerized agents can learn to generalize without having a compositional language, at least using these metrics for compositionality.

Another idea, employed by Steinert-Threlkeld [27] and Guo et al. [8] is to identify a specific characteristic whose presence would demonstrate compositionality and then perform an analysis of the emergent language to determine whether this characteristic is present. Steinert-Threlkeld [27] asserts that function words are a paradigm example of non-trivial compositionality. Since trivial compositionality means that each symbol maps to a primitive attribute, and function words such as "most" or "and" have no coherent meaning without being combined with another term, their meaning in context cannot be explained by trivial compositionality. He goes on to design the Extremity Game, where objects in the context

are purposefully chosen so that they each have the most or least extreme value of some attribute. This experiment is motivated by the goal of finding a language with extreme adjectives like "most" and "least". To evaluate, he analyzes the resulting language in a specific way to accept or reject his hypothesis of how the language will be formed. He measures how frequently the first symbol is used to indicate the attribute of the target object whose polarity distinguishes that object from the context, and he measures how frequently the second symbol is used to indicate the polarity of that attribute for that object. This gives a clear answer to whether the language is compositional in the expected way, but it is limited because it relies on accepting or rejecting a very specific hypothesis regarding the emergent language.

Guo et al. [8] use a similarly motivated evaluation approach of closely analyzing a language against a hypothesis. Whereas Steinert-Threlkeld focused on the emergence of gradable adjectives, Guo et al. study the emergence of counting words through a new signaling game called the Bag-Select Game. In this game, each object in the context is, conceptually, actually a bag of objects. They define a certain number of object types that is static across all bags, and then for each bag they choose a number of instances of each object type. These bags are represented in one of three different ways. First, they can be represented as concatenated one-hot vectors. Each object type is represented as a one-hot vector, where the position of the one corresponds to how many of that object type are in the bag. They can also be represented as a bag of one-hot vectors, where each one-hot vector represents an instance of an object of a certain type. In this setting there are a variable number of one-hot vectors depending on how many total objects are in a given bag. Finally, bags can be represented by images depicting the objects in a given bag. To evaluate the resultant languages, they primarily use `topsim` but they also construct a table for a few languages to show which symbols correspond to each possible bag (For an example, see Table 5.4). With this table it is easy to see whether a certain symbol refers to the same number of instances of the same object type in each case. Although they find highly compositional languages, this visual aid shows that number words do not exactly emerge. Similar to the analysis

of Steinert-Threlkeld [27] this targeted analysis gives a clear picture of the structure of the resulting languages, but it is limited in its ability to apply to more complex experiments or more general compositionality.

Another evaluation method to consider is presented by Lan et al. [18]. In this paper, they use the Extremity Game of Steinert-Threlkeld and evaluate the compositionality of resultant languages using a probe which they call a "composition network." The probe approach is reminiscent of generalizability and hinges on the assumption that if the agents have seen two symbols in different contexts then they should be able to understand the meaning of those two symbols juxtaposed. As an example, if the model has seen messages corresponding to *attribute a* and *polarity x*, *attribute a* and *polarity y*, and *attribute b* and *polarity x*, it should be able to understand a message corresponding to *attribute b* and *polarity y*. To evaluate this, they train a multi-layer perceptron to produce the fourth message given messages one, two, and three, then using held-out data they build the fourth message, feed it to the receiver agent, and examine the communicative accuracy. Intuitively, this seems like a promising approach, but it is a relatively unproven method that could be said to suffer from some of the drawbacks mentioned previously, namely that it is somewhat dependent on the specific experiment and that generalizability may not be a good stand-in for compositionality.

A final metric to consider, introduced by Andreas [1], is called Tree Reconstruction Error, or TRE. This is an approach that relies on finding an explicitly compositional messaging system for a given set of inputs, and then measuring how well the actual messaging system approximates this explicitly compositional one. After introducing the procedure for calculating TRE, Andreas goes on to show that it correlates with human judgements in a selected task. This is a promising metric because it seems to match intuitive judgements and apply across different experiments, though it does rely on specifying a composition function and at this time has not widely been adopted by the community.

Korbak et al. [16] do not introduce a metric for measuring compositionality, but they methodically explore each of the metrics discussed previously and ask whether they capture trivial and non-trivial compositionality. In order to answer this question, they imagine a

prototypical signaling game with objects defined by the two attributes shape and color. Then, they carefully construct nine language protocols that can identify each object in the object set. One of these languages exhibits trivial compositionality, one language is holistic, and one is randomly generated. The remaining six each embody a specific type of non-trivial compositionality. The authors then use a variety of common quantitative metrics to measure compositionality, including the ones discussed previously in this section. They find that TRE is the only metric that assigns high scores to all of the non-trivially compositional languages. This is a fascinating result, showing both the shortcoming of many common metrics used in this research, and showing the promise of TRE for capturing both trivial and non-trivial compositionality.

2.3 Factors Affecting Emergent Language Strucutre

In recent work, a number of factors have been explored with an eye to their effect on the structure of emergent languages, with particular interest in compositionality. These factors can be categorized in a few ways. One way to look at them is their effectiveness in eliciting compositionality. From this point of view, the most exciting factors are those that can be reliably shown to encourage compositionality in emergent languages. Another way is to think about their applicability to human language evolution. In this line of thought, researchers are looking for factors that may have played a role in guiding the structure of human language across the evolution of our species. Factors that reliably lead to compositionality but cannot be generalized beyond a specific set of computational experiments are then less interesting. In the following sections I will discuss iterated learning, size of communication channel, curriculum learning, and game structure with a focus on their effect on compositionality and their potential to be generalized to apply to language evolution.

2.3.1 Iterated Learning

Iterated learning is a factor that both has a good basis in language evolutionary theory and has been shown to encourage compositionality in computational experiments. Kirby et al.

[15] discuss the idea that compositionality, or linguistic structure more generally, results from the competing pressures of compressibility and expressivity. The authors go on to propose iterated learning as a manifestation of a compressibility pressure.

Iterated learning is a procedure to facilitate multi-generational simulations by including parent-to-child learning in addition to interactive training. Ren et al. [24] define a concrete process for applying iterated learning with neural models in an emergent communication setting. Their model consists of three phases, a learning phase, an interacting phase, and a transmitting phase. In the learning phase, newly initialized agents learn the language of their parents. The child sender is trained on *(input, message)* pairs from the parent sender. Given an input, the child sender produces a sequence of distributions across possible symbols and is trained using cross-entropy loss against the message that the parent produced for that input. The child receiver is trained by interacting with the newly trained child sender while the child sender's parameters are held static.

In the interacting phase, the sender and receiver from the same generation play a standard signaling game and are trained based on the receiver's choice. The receiver is trained with backpropagation and the sender is either trained with reinforcement learning or if Gumbel-Softmax [11] is employed then the whole sender-receiver system can be trained with backpropagation.

In the transmitting phase, *(input, message)* pairs are saved to be used in the learning phase of the subsequent sender. Objects from the input set are shown to the parent, and the message the parent produces is saved with each corresponding input. Ren et al. sample messages from the parent in a stochastic way, so that there is not a singular mapping from input to message.

There have been some exciting results stemming from the use of iterated learning as a pressure for compositionality. Ren et al. [24] use a basic signaling game with their neural iterated learning framework, and consistently find an increase in topographic similarity across generations, using negative cosine distance as the measure for object distance.

Guo et al. [8], who introduced the Bag-Select Game, show another example of iterated

learning maximizing compositionality. The authors closely follow the architectural design of Havrylov et al. [10] and the neural iterated learning procedure of Ren et al. The sender and the receiver are each an LSTM. The input to the sender is a single object, or bag, which is processed through a module specific to the input type (concatenated one-hot vector, bag of one-hot vectors, or image) before being used to initialize the sender LSTM. The sender produces a message of a fixed length, which is then fully processed by the receiver. In order to get a distribution across the context, the dot product is taken between the final output of the receiver and the representation of each bag in the context. Guo et al. use two object types with up to five objects of each type in each bag. Finally, they constrain messages to be length two and use a vocabulary of 10 symbols. The authors study this game across many generations and use `topsim`, employing hamming distance for object distance, to measure the compositionality of the language that is saved in phase three of each generation. Their results support the importance of iterated learning as a compressibility pressure to encourage compositionality by showing an increasing `topsim` score across generations. They are also able to show through a qualitative analysis of one resulting communication system that a specific symbol in a specific position reliably refers to a number of a specific object type. For example, w reliably communicates the meaning *0 of object A* and q reliably communicates *0 of object b*. The tight pairing between symbol and number of a certain object is very interesting to see, and certainly reflects a type of compositionality. However, it must be emphasized that they do not exactly find number words emerging. It would perhaps be more interesting to find messages where a single symbol corresponded consistently to a number word, no matter which object type it was describing.

Chaabouni et al. [5] do not specifically use a compressibility pressure in their experiments in the form of iterated learning. Instead, they run many experiments which produce a wide variety of languages, and then measure the learnability of each of the resultant languages. They show that languages which are more compositional, measured with `topsim` and `posdis`, are easier for a new agent to learn. Although this finding does not directly support the importance of iterated learning, the results do add evidence to the importance of learnability

more generally as a factor that encourages compositionality.

Complexities of Iterated Learning

Iterated learning is still relatively new in Emergent Communication, and there is not a completely standard way to implement it. Ren et al. [24] provide a good general procedure to follow, but there are still a few things that can vary between implementations.

First is the question of how to collect messages for input objects in phase 3 of the training, the transmitting phase. Ren et al. and Guo et al. both sample messages in a stochastic way. This means the parent sender may not produce the same message each time it sees a given target object. Thus, it is impossible for the child sender to learn a one-to-one mapping between object and message. But stochastic sampling is not the only choice. Carcassi et al. [4] explore whether iterated learning could be a factor that encourages the semantic universal of monotone quantifiers. Though these authors are not using the signaling game framework, their implementation of iterated learning is just as relevant. In this implementation, the authors let the sender produce messages in a deterministic way in the transmitting phase, meaning that there is a single message corresponding to each object. It then could be possible for the child to memorize an exact mapping from object to message, though this can be prevented by limiting the number of samples that are saved in the transmitting phase and by limiting the number of epochs in the learning phase for the new agents. Throughout this thesis, I will refer to the approach of Ren et al. as *stochastic transmission*, and I will call the approach of Carcassi et al. *deterministic transmission*.

2.3.2 Curriculum Learning

Curriculum learning is a factor that has not been explored in much depth in either language evolution or emergent communication, though it has shown some initial promise in encouraging a regular structure. Curriculum learning, formally introduced to the machine learning community by Bengio et al. [2], describes an approach wherein training progresses through different stages of learning that gradually increase in complexity. By learning simple things

first, the idea is that learning more complex things should then be easier. This approach has been effective in domains such as image classification [9] and natural language understanding [30], and has only begun to be explored in emergent communication. Tomlin and Pavlick [29] explore the emergence of grounded communication using a multi-round reference game called Task and Talk, where a grounded communication system is a system that has a one-to-one correspondence between referent attributes and dialogue tokens. They find that curriculum learning does help to encourage grounded communication systems.

Curriculum learning has not been explicitly proposed as a factor influencing the evolution of human language structure, but I find it plausible that early humans began communicating about simple concepts before moving on to more complex concepts. I will not argue in this thesis that curriculum learning definitely played a role in human language evolution, but I do believe that the possibility exists. Curriculum learning can then be considered a generic pressure similar to iterated learning, the difference being that iterated learning has already been shown to encourage human-like communication systems in a certain way, i.e. in their compositionality, while curriculum learning has not as conclusively shown a similar effect.

2.3.3 Size of Communication Channel

The size of the communication channel, referring to vocabulary size and message length, has been proposed as a factor that could have encouraged grammar to emerge during human language evolution [22]. However, in emergent communication experiments it has not been shown to have the effect of encouraging grammar or compositionality. The idea would be that if vocabulary and message length are limited so that there cannot be a unique primitive element in the message space for every concept in the meaning space, then composition must emerge in order to account for all necessary meanings. However, Chaabouni et al. [5] found that in order for a sender and receiver to learn to communicate effectively, the channel size must be large enough to express the whole input space. This suggests that forcibly restricting the channel, either with vocabulary size or message length, will not work.

Instead of absolutely restricting the capacity, another idea is to add a softer pressure

towards compression. Luna et al. [21] do just this by introducing something called *vocabulary loss*, which encourages the sender to make high-confidence choices with the symbols it produces, and results in a compression pressure on the used vocabulary. They argue that since the *end-of-sentence* symbol is part of the vocabulary, a smaller vocabulary also encourages shorter messages, a claim which is backed up by their results. However, they introduce no explicit pressure on message length. They also do not study the effect of this pressure directly on compositionality, but only in conjunction with other pressures.

Another example of a soft pressure on message length is explored by Rita et al. [25]. In this experiment, the authors introduce what they call *lazy senders* and *impatient receivers*, which means a sender that is incentivized to keep messages short and a receiver that is incentivized to make its choice as early as possible. Similarly to how Luna et al. implemented their vocabulary pressure, Rita et al. implement these pressures by adjusting the loss function. To operationalize the *lazy sender* they add a penalty for message length. The *lazy receiver* is slightly more complicated. They force the receiver to make a prediction after reading each symbol of the sender's message, then they take the cross-entropy loss of the receiver's prediction compared to the correct choice at each timestep. This means that the sooner the receiver chooses correctly, reducing the cross-entropy loss at that and subsequent steps, the lower the overall loss will be. They find that this novel loss successfully encourages more efficient messages, measured in a few ways, but only if they use a more standard cross-entropy loss to begin and introduce the new loss later in training. They do not study the effect of this loss on compositionality.

2.3.4 Game Structure

Game structure is different from iterated learning, curriculum learning, and compression of the communication channel in that it is a necessary part of every experiment. However, it is still an important factor to consider when discussing and making claims about results. If the game or the model is structured in such a way that compositionality is forced, then claims about general pressures on compositionality are much weaker. However, a game to explore

compositionality must be structured so that developing a compositional system is a natural way to succeed at the game.

Guo et al. [8] aim to explore compositionality from the standpoint of communicating about numeric concepts. This means that they need to have a game that involves numeric concepts, which they actualize as objects in a bag. As previously discussed, there are multiple ways to represent these bags, and two out of the three ways that they explore yield compositional languages using iterated learning. This suggests that iterated learning's effect on compositionality is conditional on the way that inputs to the model are represented. This does not detract from the importance of their findings, but it is something that must be considered.

Steinert-Threlkeld [27] finds that compositionality emerges after making a few modifications to the basic game. As discussed above, this series of experiments was based around the Extremity Game where there are a set of gradable properties or attributes in the object space. Each object in the context has either the most or least extreme values for each of those attributes. The message length is limited to two symbols, and furthermore, there are two distinct vocabularies, one for the first symbol and one for the second. The first symbol is selected from a vocabulary the size of the number of attributes in the object space. The second symbol is selected from a vocabulary of size two, corresponding to the two polarities *most* and *least*. The sender is a multi-layer perceptron and the experiment is repeated with two different types of receivers. The basic receiver is a multi-layer perceptron that takes the context and the message and produces a distribution over target objects. Steinert-Threlkeld also introduces an *attentional receiver*, which uses an attention mechanism to "choose a dimension to attend to". This forces the receiver to only look at a single attribute when it considers which of the target objects to choose. Steinert-Threlkeld finds that with the attentional receiver the agents tend to find an optimal communication protocol, very reliably with objects that have one and two attributes, and somewhat less reliably when the objects have three attributes. An analysis of one of the successful protocols for three-attribute objects shows that the symbols in the first message slot overwhelmingly tend to communicate the

relevant attribute and that the symbols in the second message slot communicate the polarity of that attribute. This shows a convincing example of non-trivial compositionality emerging, however it does so only by constraining the receiver architecture and message space, instead of by applying a more general type of pressure.

Chapter 3
METHODOLOGY

This chapter introduces a new game, called the Quantifier Game, explains the independent variables that are modified between experiments in order to explore effects on certain compositionality metrics, describes the architecture of the model used for the series of experiments, and introduces a new compositionality metric designed explicitly to look for quantifiers.

3.1 The Quantifier Game

Inspired by the approach of Steinert-Threlkeld [27], who proposed function words as case studies of non-trivial compositionality, I aim to explore the ability of neural models to develop compositionality in the form of quantifiers. To do this, I introduce the Quantifier Game. This game could be thought of as a modification of the Bag-Select Game of Guo et al. [8] but the modifications are substantial enough that the resulting game warrants a new name.

Just like Guo et al., I imagine each possible target object to actually be a bag of objects, and I use the one-hot vector representation for these bags. In this setting, if there are m object types with up to n instances of each object in each bag, then each bag is represented as a concatenation of m vectors, each of length *n+1*. The index of the 1 in each vector indicates how many of the given object type are in the bag. The length is $n + 1$ to allow for 0 of the given object to be in the bag. As an example, imagine there are 2 object types in the game (say squares and circles) and a maximum of 5 of each object in each bag. In this setting, $m = 2$ and $n = 5$. For a bag with 0 squares and 4 circles, the one-hot vectors would be $[1, 0, 0, 0, 0, 0]$ $[0, 0, 0, 0, 1, 0]$ with squares arbitrarily presented first.

The main innovation that I make with the Quantifier Game is how the target and context

bags are generated. This innovation was motivated by a consideration of the meaning of *a few* and *a lot* in various contexts. My interpretation of these quantifiers could be said to be a modification of the *cardinal* reading described by Partee [23]. Partee describes two readings of the quantifiers *many* and *few*, one cardinal and one proportional. For the cardinal reading, the set described by each quantifier is interpreted in relation to a number n. For *many*, n is usually understood to be reasonably high, and the set in question has more members than n. For *few*, n is understood to be reasonably small, and the set has fewer members than n. I imagine the n from the discussion of *few* and the n from the discussion of *many* to be on the same scale, and indeed to be bounds on a central range of numbers that are neither *a few* nor *many*. I also use *a lot* instead of *many* because I find it more natural when the noun phrase (e.g. *a lot of squares*) is not joined to a verb phrase. I realize that this interpretation ignores the fact that in English the value of n is often dependent on context, but I find my interpretation to be at least close to the cardinal reading described by Partee and this interpretation makes the quantifiers simpler to model. For the rest of this thesis, I will use *a few* to mean less than a certain central range, and *a lot* to mean more than a certain central range. I could just as well introduce new quantifiers that have this meaning, but I find it easier to use existing quantifiers whose meaning is at least close to this.

In the original Bag-Select Game, every bag in the context is generated by randomly picking the number of each object type o in the bag and then randomly picking one bag to be the target. In contrast to this, I carefully define how the bags are generated. Given a maximum number of each object type per bag, n, the object space is split into three sectors in the following way:

> Pick a size, s, that is less than half of n, which is constant for the whole experiment.
> The low sector consists of the range 0 to s
> The middle sector consists of the range $s + 1$ to $n - s - 1$
> The high sector consists of the range $n - s$ to n

As a concrete example, imagine n is 9 and s is 3. The low sector would be 0–3, the middle sector would be 4–5, and the high sector would be 6–9. I claim that an example of

quantifiers would be symbols that denote these sectors, something akin to *a few* for the low sector, and *a lot* for the high sector.

In order to encourage the model to develop quantifiers, I further define the way that target bags are chosen such that they all satisfy the following constraints: Pick a target sector x, which is either *high* or *low*. Pick a target object type o. Generate a bag where the number of o falls into sector x. No other object types in this bag may have a number that falls into sector x. Additionally, given the opposite sector x', there must be either 0 or more than 2 object types whose number falls into x'. This simplifies the search for quantifiers during analysis by making it clear which quantifier the sender *should* use if it is using quantifiers in its communication protocol. If we consider a setting with 3 object types, *squares*, *circles*, and *triangles*, a valid bag would be *(2 squares, 4 circles, 4 triangles)*. The amount of squares falls uniquely into the low sector, and circles and triangles both fall into the middle sector. An invalid bag would be *(2 squares, 2 circles, 4 triangles)*. This puts squares and circles in the low sector, and triangles in the middle sector. This is a violation because neither the high nor the low sector has a unique shape. Another invalid bag would be *(2 squares, 4 circles, 7 triangles)*. Here, squares are alone in the low sector and triangles are alone in the high sector. This is a violation, because there is more than one possible way to describe this bag using quantifiers.

The context, which is what the receiver sees along with a message from the sender, consists of the target bag and a set of distractors. Candidate distractor bags are generated the same way as the target bags, so that the receiver cannot succeed simply by learning what a target bag looks like. There is also an additional constraint on how the distractor bags are selected. When the target bag has only 1 object type whose number falls into sector x, then there can be no other bags in the context with only 1 object type in sector x. For example, if the target bag has *(0 squares, 7 circles, 8 triangles)* then there cannot be a bag in the context with *(1 square, 5 circles, 5 triangles)*, because then there would be two bags with a uniquely low amount of squares.

Using less precise language, and understanding *a few* and *a lot* to denote the low and

high sectors in the number space, the game can be described in the following way. Each bag has either a few or a lot of a certain object, and only that object. If the target bag has a few of a given object, then it is the only bag in the context with a few of that object. A target bag cannot have a few of a single object and have a lot of a single other object. Then the agents could succeed by learning a language with messages analogous to *pick the bag with a few of object a* or *pick the bag with a lot of object b*.

3.2 Additional Factors to Encourage Quantifiers

The design of the quantifier game makes it such that one successful way to communicate is by using quantifiers. However, there are also other successful communication systems that can be developed. One of my aims is to explore whether there are factors that make the neural model more likely to develop quantifiers. Since quantifiers are a case study in non-trivial compositionality, I plan to use factors that have been proposed as pressures for compositionality both in emergent communication specifically and in the evolution of human language. The factors that I will explore are restrictions on the communication channel, curriculum learning, and iterated learning.

3.2.1 Restrictions on the Communication Channel

One of my questions is whether a pressure to compress the communication channel will increase compositionality. The size of the communication channel is a combination of vocabulary size and message length. I leave vocabulary fixed while introducing a pressure to reduce message length in the form of a novel term in the loss function. When the sender produces a message, each symbol is chosen by forming a distribution across the vocabulary using softmax and then sampling. In every one of my experiments, I allow for variable length messages by including an *end-of-sentence* (`eos`) symbol. If the sender produces the `eos` symbol at position `n`, then the message transmitted to the receiver consists only of the symbols at positions `0` through `n-1`. The term added to the standard loss is a modified cross-entropy loss focusing on the symbols emitted by the sender, and can be written as equation 3.1.

$$\lambda \sum_{t=0}^{|T|} \log p_t(\texttt{eos}) \tag{3.1}$$

In equation 3.1, t ranges over symbols in an emitted message, and $|T|$ is the length of the emitted message. $p_t(eos)$ is the probability that the sender assigns to the `eos` symbol for a given step in the message transmission. I call λ the *length lambda*, which is used to scale the impact of this message length loss in relation to the standard loss. In essence, this is encouraging the sender to produce the `eos` symbol at every step. This term is then added to the standard loss and the sum is used for gradient computation. I perform a series of experiments varying the value of the length lambda while asking two questions: will this new message length loss successfully constrain message length without affecting accuracy, and will this loss have an effect on compositionality?

3.2.2 Curriculum Learning

The basic quantifier game was described above. In the curriculum learning setting I consider that basic game to be stage 2 of training, and I introduce a modified game for stage 1 of training. For the first stage, each target bag will only contain a single object type. The object type will be randomly chosen, and the amount of this target object type in the target bag will also be randomly chosen. The receiver will see a context the size of the number of total object types. If there are 3 object types (e.g. squares, circles, triangles), then the context size will be 3. Each bag in the context will have a nonzero amount of one object, and zero of all others, with one bag for every object type. The game is successful if the receiver picks the bag from the context that contains the same object type as the target bag seen by the sender. Note that the exact bag seen by the sender may not be in the context. The point of the game at this stage is to recognize object types, not numbers. In simpler language, if the sender sees a bag that contains a non-zero amount of triangles, the receiver should pick the bag from the context that also contains any non-zero number of triangles.

After the two agents reach convergence playing stage 1 of the game, they transition into

stage 2, which is the basic quantifier game. My hypothesis is that this will encourage a system of quantifiers wherein the agents first learn to communicate about object types by mapping types to symbols, and then learn to communicate about quantities by mapping quantities to symbols.

3.2.3 Iterated Learning

Since iterated learning has shown so much promise in encouraging trivially compositional languages, I want to explore its effectiveness at encouraging nontrivial compositionality in this case study of quantifiers. As discussed previously, while iterated learning is a straightforward concept, there are complexities to consider in an implementation.

First, one must decide which agents will learn from the parents: the sender, the receiver, or both. Ren et al. [24] conclude that training the child receiver does not have a significant effect on language structure, so to minimize complexity I will only train the child sender using its parent's language.

The next thing to decide is whether to use the deterministic or stochastic approach in the transmitting phase, when object-message pairs (or bag-message pairs) are saved in order to train the next child sender. Both of these approaches have shown promise in different settings, so I will experiment with both.

Whether using deterministic or stochastic transmission, there are a few other complications to consider, namely, how many samples to collect in the transmitting phase, and how many epochs to allow for the child learning phase. These two variables control how much of the parent's language the child is exposed to and how well the child is able to learn the subset of the language that it sees. Using stochastic transmission, the number of the samples should be higher than the total number of unique objects. Guo et al. [8] choose a number that is many multiples of the total number of unique objects in order to give the child a view of the full distribution of target-message pairs in the parent's language. Using deterministic transmission, the number of samples cannot be more than the number of unique targets, and it makes the most sense to have the number be smaller than the number of unique targets

so that the child sender cannot completely learn the parent's language.

For my study, I perform a series of experiments using both deterministic and stochastic transmission while varying the number of samples the child sees and the number of epochs of child learning. In the stochastic condition I only vary the number of samples while leaving the number of epochs at one with the thought that as the number of samples gets large enough there are sure to be repeated target-message pairs, so this is effectively equivalent to having a smaller number of samples with more epochs.

3.3 Architecture

The model I use has essentially the same structure as the model used by Guo et al. [8] and Havrylov and Titov [10]. It consists of two agents, a sender and a receiver, each of which is an RNN, specifically an LSTM.

For the sender, the hidden state will be initialized with an encoded bag representation. At each timestep t, the LSTM will take as input the previously emitted symbol s_{t-1}, passed through a linear symbol embedder, and will produce a symbol s_t. The symbol is produced by passing the output of the LSTM through a linear transformation so that it has the same size as the vocabulary. The values coming out of the linear tranformation are put through a softmax layer and then the symbol is sampled from the resulting distribution. The symbols are taken from an arbitrary vocabulary, and the symbol set includes an end-of-sentence symbol.

The receiver is initialized randomly, and at each timestep it takes as input the next symbol in the sender's message, passed through the linear symbol embedder. After processing the entire message, a dot product is taken between the final receiver output and the representation of each bag. If the sender produced the end-of-sentence symbol, then we consider the receiver output produced just before seeing this symbol to be the final output. These dot product values are put through a softmax layer to form a probability and then a sampling operation is done to indicate the receiver's prediction. For evaluation, all sampling operations are replaced with greedy operations, taking the maximum value from the distribution.

The loss function is a sum of the standard cross-entropy loss between the receiver's probability distribution and the correct target choice added to the novel message length loss. Gradients are computed with backpropagation using the Gumbel-Softmax estimator [11] where the sender's sampling operation would take place. Adam [13] is used as an optimizer, and full hyperparameter settings are provided below.

3.4 Evaluation

I plan to use two quantitative evaluation metrics and a qualitative inspection inspired by trends in Emergent Communication literature. The most common metric proposed for measuring compositionality is topographic similarity, which measures how closely the distance between two objects matches the distance between corresponding messages. An important choice when using `topsim` is how to measure distances. In the message space, Levenshtein distance is fairly standard and I plan to follow this practice. In the object space, common measurements are hamming distance, Euclidean distance, and cosine distance. Guo et al. [8] use hamming distance when they analyze the Bag-Select Game. However, I find hamming distance to be a slightly unnatural choice for this setting. With the concatenated one-hot vector representation, every bag will have a single 1 for every object type. This means that the maximum hamming distance is two times the number of object types. Additionally, hamming distance does not capture the intuition that some numbers are closer than others. A bag that has 3 squares and 4 circles seems closer to a bag with 2 squares and 5 circles than to a bag with 1 square and 1 circle. Hamming distance would treat these three bags as equidistant. Instead, I choose to use Euclidean distance, which seems to more closely match intuition. To calculate `topsim` in my experiments, I use the core library from the EGG package of Kharitonov et al. [12].

In addition to `topsim`, I introduce a novel metric called Attribute Score, or `attsco`, to look for quantifiers specifically. This metric is inspired by `posdis` and `bosdis` of Chaabouni et al. [5], but varies in an important way. `Posdis` iterates over positions in the message, and at each position looks for an attribute that has high mutual information with this position

as compared to all other attributes. Then the average over each position is considered the `posdis` score. `Attsco` focuses first on attributes instead of position. `Attsco` has two variants, *positional* and *symbol-wise* Attribute Score.

Positional Attribute Score, or `posattsco`, can be calculated with equation 3.2.

$$posattsco_i = \max_j \frac{I(a_i, p_j)}{H(a_i)} - \frac{I(a_2, p_j)}{H(a_2)} \tag{3.2}$$

In this equation, a_i stands for the i-th attribute in the object space and p_j stands for j-th position in a message. a_2 stands for the attribute, excluding a_i, that has the maximum mutual information with p_j. What the equation aims to do is this: for a given attribute, find the position that has the highest exclusive mutual information with that attribute, that is, the position that has more mutual information with that attribute than any other. For that position, find the difference of its mutual information with the target attribute and the attribute with the next highest mutual information. Normalize these values by attribute entropy to get a score between 0 and 1.

Symbol-wise Attribute Score is calculated in a similar way, just using symbols in the message space instead of positions. `Attsco` yields a score for a chosen attribute, but a truly compositional system should have a high `attsco` for every attribute in the meaning space. This can be captured by taking an average `attsco` across all attributes. This average will be maximized in a language in which each attribute has a corresponding position or symbol that communicates a maximum amount of information for that attribute alone.

I will use this `attsco` metric in order to calculate something I call Quantifier Score. This is done by transforming the object space in such a way that one of the attributes corresponds to the quantifiers that I have defined. Recall that the target bags in the quantifier game each have a single object type whose amount falls uniquely into a given sector, either low or high. In other words, There are uniquely *a few* or *a lot* of a certain object type in each bag. This bag can be described using the attributes *object type* and *sector*. The values of the *object type* attribute correspond to the set of unique object types used in the game, and the values of the *sector* attribute are *low* and *high*. Quantifier score is the `attsco` calculated using the

sector attribute. This is in essence a way to measure non-trivial compositionality by using a metric designed for trivial compositionality while performing a *non-trivial* transformation of the object space.

Finally, in addition to `topsim` and quantifier score, I plan to perform a qualitative examination of a few of the resultant communication protocols to look for any patterns not captured by the metrics that show evidence for or against compositionality.

3.5 Experiments

For all of my experiments, each LSTM has a hidden size of 64, the encoder has a hidden size of 128, the dimension of the symbol embedder is 64, the vocabulary size is 20 including `eos` and the maximum message length is 10.

Every experiment uses three object types and up to eight objects of each type. I split the number space into three even sectors, with low encompassing 0-2, middle 3-5, and high 6-8. The quantifier game with three types and up to eight objects per type results in a total of 324 possible target bags. I use a training set of size 260, a validation set of size 32, and a test set of size 32. Each experiment has a context size of 5, where every bag in the context is taken from the same split.

I experiment with the pressures of message compression, curriculum learning, and iterated learning. For message compression, I experiment with length lambdas of .00001, .00005, .0001, .0005, and .001. I found that any numbers higher than that resulted in the models being unable to arrive at a successful communication protocol.

For iterated learning, I experiment with deterministic and stochastic transmission phases. For deterministic transmission, I choose 10, 20, 40, 80, and 160 *(target, message)* pairs, and I run the child learning phase for 50, 250, and 500 epochs. For stochastic transmission, I choose 100, 400, 1600, 3200, 6400, 12800, and 25600 samples, all with 1 epoch for child learning. Sender training uses a minibatch size of 8.

For the interactive training phase, I set minibatch size to 16, and set the maximum number of rounds to 100,000. However, I use early stopping based on validation loss and

training never reaches the maximum number of rounds.

I run a series of trials for the following settings:

Basic game with no compression

Basic game with compression

Curriculum learning with no compression

Curriculum learning with compression

Iterated learning with no compression.

I run 10 trials of each of the non-iterated learning settings. Since iterated learning is more time and compute intensive, I run 5 repetitions of each iterated learning setting, except for the 50 epoch variants of the deterministic setting where I run 10 repetitions. The repetitions of the same setting have a different random seed but leave everything else the same.

Chapter 4

RESULTS

The results chapter is divided into a few sections to show overall results across the 5 coarse-grained settings and then more detailed results for each individual variants. For each section, I present accuracy, `topsim` using Euclidean distance in the object space, and quantifier score. I found the most significant effects using positional quantifier score as opposed to symbol-wise, so that is what is reported in all tables. All metrics reported are on the validation set. The tables all show mean and standard deviation, with some rows bolded to indicate a significant difference from the results of the basic setting. To test for significance, I used Welch's t-test since population sizes are not all equal, and I consider a p value of less that .05 to indicate significance.

4.1 Aggregate Results

Table 4.1 shows results for the 5 coarse-grained settings:

Basic game with no compression

Basic game with compression

Curriculum learning with no compression

Curriculum learning with compression

Iterated learning with no compression.

Recall that compression and iterated learning each encompass a few different finer-grained settings, and this table shows an aggregate of all of those.

Note that the accuracy of all settings is quite high, indicating that the model was able to converge on a successful communication system. `Topsim` has a mean between .3 and .5

Setting	Quantifier Score	Topsim	Accuracy
Basic, no compression	0.02 ± 0.12	0.47 ± 0.1	1 ± 0
Basic with compression	0.03 ± 0.13	0.42 ± 0.12	0.96 ± 0.14
Curriculum, no compression	0.08 ± 0.12	**0.38 ± 0.09**	0.99 ± 0.01
Curriculum with compression	0.03 ± 0.1	**0.32 ± 0.14**	0.93 ± 0.12
Iterated - all	0.05 ± 0.13	0.45 ± 0.08	0.99 ± 0.04
Iterated - stochastic	0.05 ± 0.14	0.45 ± 0.08	0.98 ± 0.07
Iterated - deterministic	0.05 ± 0.12	0.45 ± 0.09	0.99 ± 0.03

Table 4.1: Aggregate Results of all Settings. Mean ± SD, significant results in bold font ($p < .05$)

for all settings, with standard deviations between .08 and .14. There are two significant effects when considering `topsim`. For settings with no compression, `topsim` for the curriculum condition (m=.38, sd=.09) is significantly lower that the basic condition (m=.47, sd=.1). For settings with compression, we see a similar effect, with `topsim` for curriculum learning (m=.32, sd=.14) again significantly lower than `topsim` for the basic condition (m=.42, sd=.11). Quantifier score has a mean below .1 in all cases, with quite high standard deviation relative to this mean. There are no significant effects on quantifier score between settings.

4.2 Compression

Table 4.2 shows results for the basic setting compared to each of the compression settings. The leftmost column indicates the value of the length lambda, where a higher value corresponds to a higher compression pressure. I have also included a row showing the aggregate of all compression settings. The only significant effect we see is with length lambda .001, which yields a `topsim` of (m=.29, sd=.14), significantly lower than the basic setting with no compression (m=.47, sd=.1). Notice that the accuracy mean is also lower in this setting (m=.86, sd=.25) but the difference is not significant (p=.1).

Length Lambda Value	Quantifier Score	Topsim	Accuracy
0	0.02 ± 0.12	0.47 ± 0.1	1 ± 0
All non-zero lambdas	0.03 ± 0.13	0.42 ± 0.12	0.96 ± 0.14
0.00001	0.09 ± 0.15	0.44 ± 0.05	1 ± 0
0.00005	-0.04 ± 0.07	0.46 ± 0.06	1 ± 0.01
0.0001	0.01 ± 0.12	0.5 ± 0.09	0.95 ± 0.14
0.0005	0.09 ± 0.18	0.39 ± 0.12	1 ± 0.01
0.001	0.01 ± 0.05	**0.29 ± 0.13**	0.86 ± 0.25

Table 4.2: Fine-grained results for compression settings. Rows are organized by values of length lambda. Bolded font indicates significant difference from basic setting with length lambda of 0.

4.3 Curriculum Learning

Table 4.3 shows the detailed results for curriculum learning. Curriculum learning was used in conjunction with compression in order to look for a combined effect. In the leftmost column the header rows show the value of length lambda, corresponding to compression pressure, and under each header row there is a row for the basic setting with that lambda value and the curriculum learning setting with that lambda value. For length lambda of 0, `topsim` for the curriculum condition (m=.38, sd=.09) is significantly lower than basic (m=.47, sd=.1). For length lambda of .00005 and .0001 we see a similar effect, with curriculum learning yielding a significantly lower `topsim` than basic. There are no significant effects on quantifier score or accuracy.

4.4 Iterated Learning with Stochastic Transmission

Table 4.4 shows results for each variant of the stochastic iterated learning condition. The rows show the basic setting with no iterated learning, an aggregate of all stochastic settings, and then a row for each of the numbers of samples used for child learning. Here we see

Setting	Quantifier Score	Topsim	Accuracy
Length Lambda 0			
Basic	0.02 ± 0.12	0.47 ± 0.1	1 ± 0
Curriculum	0.08 ± 0.12	**0.38 ± 0.09**	0.99 ± 0.01
Length Lambda .00001			
Basic	0.09 ± 0.15	0.44 ± 0.05	1 ± 0
Curriculum	0.03 ± 0.13	0.35 ± 0.16	0.97 ± 0.05
Length Lambda .00005			
Basic	-0.04 ± 0.07	0.46 ± 0.06	1 ± 0.01
Curriculum	0.02 ± 0.13	**0.31 ±0.19**	0.9 ± 0.15
Length Lambda .0001			
Basic	0.01 ± 0.12	0.5 ± 0.09	0.95 ± 0.14
Curriculum	0.02 ± 0.1	**0.33 ± 0.12**	0.98 ± 0.05
Length Lambda .0005			
Basic	0.09 ± 0.18	0.39 ± 0.12	1 ± 0.01
Curriculum	0.03 ± 0.08	0.32 ± 0.12	0.9 ± 0.15
Length Lambda .001			
Basic	0.01 ± 0.05	0.29 ± 0.13	0.86 ± 0.25
Curriculum	0.03 ± 0.08	0.29 ± 0.12	0.91 ± 0.15

Table 4.3: Results for curriculum learning compared to basic, organized by length lambda. Bolded font indicates significant effect between curriculum and basic for a given length lambda value

Samples	Quantifier Score	Topsim	Accuracy
Basic	0.02 ± 0.12	0.47 ± 0.1	1 ± 0
All stochastic	0.05 ± 0.14	0.45 ± 0.08	0.98 ± 0.07
100	0.06 ± 0.19	0.46 ± 0.07	1 ± 0
400	0.06 ± 0.09	0.44 ± 0.11	0.99 ± 0.01
1600	**0.19 ± 0.1**	0.44 ± 0.03	1 ± 0
3200	0.14 ± 0.11	**0.36 ± 0.05**	**0.88 ± 0.16**
6400	-0.02 ± 0.06	0.44 ± 0.12	0.99 ± 0.03
12800	-0.02 ± 0.18	0.51 ± 0.06	1 ± 0.01
25600	-0.05 ± 0.13	0.5 ± 0.05	1 ± 0

Table 4.4: Results for Stochastic Iterated Learning Settings. Bold font indicates significant difference from basic setting.

one significant effect on quantifier score, with the 1600 sample setting of iterated learning yielding a higher quantifier score (m=.19, sd=.1) than the basic setting (m=.02, sd=.12). The 3200 sample setting yields a significant negative effect on `topsim` (m=.36, sd=.05) and accuracy (m=.88, sd=.15) compared to basic (topsim m=.47, sd=.1, accuracy m=1, sd=0).

4.5 Iterated Learning with Deterministic Transmission

Table 4.5 shows results for each variant of the deterministic iterated learning condition. The two topmost rows show the non-iterated setting and an aggregate of all deterministic settings. The rest of the rows are grouped first by number of samples saved from the parent, and then number of epochs used for child learning. In this table we see two significant effects, both for the setting with 40 samples and 50 epochs. This iterated setting gives a significantly higher quantifier score (m=.15, sd=.1) than the basic setting (m=.02, sd=.12) and a significantly lower `topsim` (m=.38, sd=.07) than basic (m=.47, sd=.1).

Samples	**Epochs**	**Quantifier Score**	**Topsim**	**Accuracy**
Basic	Basic	0.02 ± 0.12	0.47 ± 0.1	1 ± 0
All deterministic	all deterministic	0.05 ± 0.12	0.45 ± 0.09	0.99 ± 0.03
10	50	0.05 ± 0.15	0.45 ± 0.11	1 ± 0
10	250	0.06 ± 0.04	0.45 ± 0.1	1 ± 0.01
10	500	0 ± 0.15	0.47 ± 0.08	1 ± 0
20	50	0.12 ± 0.11	0.44 ± 0.06	0.99 ± 0.02
20	250	0.03 ± 0.11	0.46 ± 0.04	0.97 ± 0.06
20	500	-0.04 ± 0.08	0.5 ± 0.06	1 ± 0
40	50	**0.15 ± 0.11**	**0.38 ± 0.07**	0.98 ± 0.06
40	250	0.13 ± 0.2	0.44 ± 0.1	1 ± 0
40	500	-0.03 ± 0.09	0.46 ± 0.03	1 ± 0
80	50	0.06 ± 0.1	0.42 ± 0.1	0.98 ± 0.06
80	250	-0.03 ± 0.06	0.48 ± 0.07	1 ± 0
80	500	-0.02 ± 0.05	0.47 ± 0.07	1 ± 0
160	50	0.06 ± 0.15	0.48 ± 0.11	1 ± 0.01
160	250	-0.04 ± 0.05	0.5 ± 0.1	1 ± 0
160	500	-0.02 ± 0.05	0.48 ± 0.06	1 ± 0

Table 4.5: Results for Deterministic Iterated Learning Settings, with rows organized by samples then epochs. Bolded font indicates significant difference from basic setting

Chapter 5
DISCUSSION

Looking at the results tables, it's clear that quantifiers are not convincingly emerging as a result of any variations of the Quantifier Game that I explored. However, there are a number of interesting observations to discuss. This chapter will examine the communication strategy used by the agents, then explore the effects of each of the three pressures, namely, message compression, curriculum learning, and iterated learning.

5.1 Communication Strategy

In all emergent communication experiments, one basic question to ask is *how are the agents assigning messages to objects?* Even if maximal compositionality is not achieved, this can still tell us something interesting about how the models behave in a certain situation. Ignoring compositionality for the moment, there are a few strategies we might expect for the Quantifier Game.

First, the sender might produce a unique message for each target bag. No matter what game is being played, this is a reliable way to succeed. Second, the sender might communicate about the sector that each object type falls into in each bag. This method depends on the agents being able to recognize that the message space has been divided in a certain way while generating bags. Finally, the sender might communicate just about the sector of the relevant object type. For instance, if the target bag has a low number of circles, a low number of squares, and a high number of triangles, the receiver should have enough information to pick the correct bag if it receives information only about the quantity of triangles.

An examination of the communication strategies that emerge show us that the agents do not always exactly use one of those three strategies, but they seem to favor the third strategy.

It is clear that there is not a unique message for every possible target bag. The total number of target bags in the validation set is 32 and the average number of unique messages across all settings taken in aggregate is 14.5 (sd 6.8). The systems also clearly don't always produce a single messages per sector and type of the relevant object. The average number of messages per *(relevant type, relevant sector)* is 2.5 (sd=1.2). However, a qualitative examination of a few communication systems show that even when this number is greater than 1, as it is in most cases, there is a distinctive similarity between messages for a given *(relevant type, relevant sector)* See Table 5.1 and notice that for target bags where the necessary information is that object type A is in the low sector, every message starts with 1 or 2 *4's* and concludes with a sequence of *h's*. Though I didn't perform this qualitative analysis for every final language, this shows that in at least some cases agents are very closely following the strategy of communicating only about the bare minimum information needed for success, namely the type and sector of the relevant object. To think about this result more generally, it is interesting to see that agents can recognize relevant attributes, even when those attributes are in a more abstract object space and not explicitly the surface attributes of the objects (or bags), and learn to communicate only about these relevant attributes.

5.2 Effects of Compression

The first question to ask about the pressure for message compression is whether the novel loss was successful in encouraging shorter messages. The answer is clearly yes, at least for a few values of the length lambda. Examine Table 5.2 to see how various lambda values affected message length. With no length lambda, the agents in the basic setting always use maximum length messages. Looking at all lambda values in aggregate, the average message length is already significantly shorter than the no compression setting (m=7.95, sd=2.84). For lambda values of .0005 (m=6.87, sd=2.04) and .001 (m=4.72, sd=2.97) this effect on message length is more pronounced. This compression happened without a significant negative effect on accuracy. This highlights the promise of this method as a more natural constraint on message length compared to arbitrarily choosing a smaller maximum length. However, more work is

Relevant Object Type, Relevant Sector	**Message**
Object A, High	gggggggggg
Object A, High	ggggg
Object A, High	gggggg
Object A, High	ggggggggg
Object A, High	gggggggggg
Object A, High	gggg
Object A, High	ggggggg
Object A, Low	4hhhhhhhhh
Object A, Low	44hhhhhhhh
Object B, High	4444444444
Object B, Low	aaaaaaaaaa
Object C, High	hhhaaaaaaa
Object C, High	hhhhaaaaaa
Object C, Low	iiiiiiiii

Table 5.1: A list of all messages from the validation split of one iterated learning run of the quantifier game, grouped by relevant object type and relevant sector.

Length Lambda	Message Length
0	10 ± 0
All non-zero lambdas	$\mathbf{7.9506 \pm 2.8376}$
0.00001	10 ± 0
0.00005	9.575 ± 0.9084
0.0001	8.5938 ± 3.0037
0.0005	$\mathbf{6.8688 \pm 2.042}$
0.001	$\mathbf{4.7156 \pm 2.9724}$

Table 5.2: Message Length for different values of length lambda, to show the effect of the compression pressure

needed to explore the best practices for using this new loss.

The next question to ask is whether this compression pressure had an effect on the structure of the communication system, in particular in terms of compositionality. The compression pressure had no significant effect on quantifier score, and for the setting with length lambda of .001 it had a negative effect on `topsim`. This is a bit of a surprising result, because the hypothesis was that a compression pressure would increase compositionality. This effect should be further studied in future work. In particular, it could be interesting to restrict message length to two, since that is the number of relevant attributes.

5.3 Effects of Curriculum Learning

The result of curriculum learning is also surprising. I hypothesized that by putting the agents through two stages of learning with different information required in each, they would map one type of information to a set of symbols or positions, and then map the second type of information to a second set of symbols or positions. However, curriculum learning yielded no improvement to quantifier score, and for a few values of length lambda it had a negative effect on `topsim`. Similar to the result for compression, this negative effect on `topsim` should

Setting	Type Score
Length Lambda 0	
Basic	0.2 ± 0.08
Curriculum	**0.31 ± 0.1**
Length Lambda .00001	
Basic	0.21 ± 0.11
Curriculum	0.27 ± 0.07
Length Lambda .00005	
Basic	0.22 ± 0.09
Curriculum	0.21 ± 0.09
Length Lambda .0001	
Basic	0.23 ± 0.09
Curriculum	0.2 ± 0.14
Length Lambda .0005	
Basic	0.19 ± 0.15
Curriculum	0.17 ± 0.1
Length Lambda .001	
Basic	0.11 ± 0.07
Curriculum	**0.2 ± 0.09**

Table 5.3: Type score reported for curriculum learning compared to basic, organized by length lambda. Bolded font indicates significant effect between curriculum and basic for a given length lambda value

be further investigated in future work.

Interestingly, if we look at type score instead of quantifier score we see a slightly different story (see Table 5.3). Recall that type score is a different application of the new `attsco` metric, focusing on relevant object type in a given target bag. For length lambda values of 0 and .001, we see a significantly positive effect on type score compared to the basic setting. This makes some sense, because stage one of the curriculum encourages the agents to focus on object type and not on sector. This result suggests that curriculum learning does have some influence over the structure of emergent communication systems, even if the results don't completely match the results that Tomlin and Pavlick [29] found studying the effect of a curriculum on groundedness. The result that I found is also somewhat limited, in that there are other ways that a curriculum might be designed besides what was done in this experiment.

5.4 Effects of Iterated Learning

Considering the fine-grained iterated learning settings, we see a significant positive effect on quantifier score in the deterministic variant with 40 samples and 50 epochs, and in the stochastic variant with 1600 samples. This tells us a few interesting things.

First, notice that these settings are neither the smallest nor the largest bottleneck settings. Considering the theoretical motivation for iterated learning and its effect on compositionality, this makes some sense. If the bottleneck is too small, and the child learns too little of the parent's language, it is as if the interactive phase starts from scratch and a completely new communication system is developed. If the bottleneck is too large, then the child can come close to memorizing the parent's language. These results support the idea that iterated learning has its best effect when the bottleneck is neither too large nor too small.

Another interesting thing to note is that we see a significant effect using both the deterministic and stochastic variants of iterated learning. At this point, there is not an agreed-upon standard for how to implement iterated learning, with some researchers using the deterministic method and some stochastic. Though this result is far from conclusive, it

	0A	1A	2A	3A	4A	5A
0B	wq	yq	uq	sq	xq	zq
1B	wy	yy	uy	sy	xy	zy
2B	ws	ys	us	ss	xs	zs
3B	wt	yt	ut	st	xt	zt
4B	wu	yu	uu	su	xu	zu
5B	wv	yv	uv	sv	xv	zv

Table 5.4: Example of fully compositional language from Bag-Select Game [8]. Columns correspond to number of shape A, and rows to number of shape B.

at least suggests that both approaches can achieve similar effects. Further exploration of different implementations of iterated learning is a ripe area for future research.

5.4.1 Why is Topsim not Maximized?

The most surprising result is the effect, or lack thereof, that iterated learning has on `topsim`. Recent work has found a very clear positive effect on `topsim` using iterated learning in a variety of experiments. My experiment finds no such effect, and in fact two settings of iterated learning show a negative effect on `topsim`, including on the same setting that showed a positive effect on quantifier score (40 targets and 50 epochs). As baffling as this seems on the surface, it can be explained with a closer examination of the Bag-Select Game of Guo et al. [8] compared to this game.

Table 5.4 shows a maximally compositional language that emerged from the Bag-Select Game. This game had two object types and up to 5 of each object type in each bag. The columns show bags with 0-5 of object A, and the rows show bags with 0-5 of object B. We can clearly see that 0 of object A is represented by the same symbol (w) in every bag, and we similarly see a consistent mapping of *(object type, object number)* pair to symbol across all bags. One way to think of this language is that there is a single symbol per *(attribute,*

Samples	Messages per relevant (type, sector) pair
Basic	2.7167 ± 1.0945
All Stochastic	2.4714 ± 1.1242
100	2.8667 ± 1.1571
400	2.6333 ± 1.2985
1600	3.8 ± 0.7853
3200	3.2 ± 0.7764
6400	1.9333 ± 0.4503
12800	**1.6 ± 0.2528**
25600	**1.2667 ± 0.0913**

Table 5.5: Number of messages per relevant (type, sector) pair for stochastic settings. Bolded font indicates a significant difference from the basic (non-iterated) setting.

value) pair, where the two attributes are shape a and shape b, and the values are the number of the given shape. Since each message communicates information about both attributes, this results in a fully compositional language, at least using hamming distance.

Now consider how the experiments I presented using the Quantifier Game differ from the bag-select experiments. Instead of communicating information about the exact number of each object type in each bag, the agents communicate about the sector that each object type falls into. In this way, we can think of a transformed attribute space where there are 3 attributes corresponding to object types A, B, and C, and 2 values for each attribute, corresponding to high sector and low sector, or *a few* and *a lot*. Another important difference is the length of the messages. Whereas with Bag-Select there are the same number of positions in the message as there are attributes, in the Quantifier Game there are many more positions. So when we look for a mapping from message to meaning, we might consider that sequences of symbols could be the smallest meaningful units, instead of single symbols. This distinction is somewhat analogous to tokens compared to characters in many written

Samples	Epochs	Messages per relevant (type, sector) pair
Basic	Basic	2.7167 ± 1.0945
All deterministic	all deterministic	2.3133 ± 1.2383
10	50	2.6833 ± 1.4064
10	250	3.1 ± 1.4414
10	500	2.1 ± 1.0179
20	50	3.4333 ± 1.301
20	250	2.5333 ± 0.5191
20	500	1.6333 ± 0.4472
40	50	3.6833 ± 0.6307
40	250	1.8667 ± 0.7491
40	500	**1.1667 ± 0.2887**
80	50	2.8167 ± 1.1586
80	250	**1.0667 ± 0.0913**
80	500	**1.2 ± 0.2739**
160	50	2.1 ± 0.9102
160	250	**1.1 ± 0.1491**
160	500	**1.0667 ± 0.1491**

Table 5.6: Number of messages per relevant type, sector pair for the deterministic iterated learning setting. Bolded font indicates a significant difference from the basic (non-iterated) setting.

languages.

Considering these differences, we might say that whereas in the Bag-Select Game, iterated learning encouraged a single symbol to always refer to a single *(attribute, value)* pair corresponding to *object type* and *number*, in the Quantifier Game we might expect a single token, or sequence of symbols, to refer to a single *(attribute, value)* pair corresponding to *object type* and *sector*.

Finally, there is one more difference to consider. With Bag-Select, in order to be successful the receiver needs information about every object type, or every attribute. Since there are two slots in every message and two object types in every bag, it is natural for a single symbol to convey information about a single object type. In the Quantifier Game, the receiver only needs information about a single relevant object type, or a single attribute. This means that it is natural for sequences of 10 symbols to act as a token and convey information about a single object type. To complete the analogy, whereas with Bag-Select, iterated learning encouraged a single symbol to refer to a single relevant *(attribute, value)* pair, with the Quantifier Game we might expect iterated learning to encourage a single token, or sequence of 10 symbols, to refer to a single relevant *(attribute, value)* pair in the transformed attribute space.

This is exactly what happens for a few settings of iterated learning. Examine Table 5.5 to see the number of messages per relevant *(object type, sector)* pair for stochastic iterated learning, and see Table 5.6 for the same information for the deterministic variants. Though we don't see the number of messages per *(object type, sector)* pair always reduced to 1 in any setting, we see a significant effect in many settings and there were a few individual runs that saw exactly 1 message per pair. For the stochastic variants, we see signficantly fewer messages per relevant *(object type, sector)* pair in the setting with 12,800 samples (m=1.60, sd=.25) and 25,600 samples (m=1.27, sd=.09) compared to basic (m=2.72, sd=1.1). For the deterministic variants, we see significantly fewer messages in the setting with 40 samples and 500 epochs (m=1.17, sd=.29), 80 samples and 250 epochs (m=1.07, sd=.09), 80 samples and 500 epochs (m=1.20, sd=.27), 160 samples and 250 epochs (m=1.10, sd=.15), and 160

samples and 500 epochs (m=1.07, sd=.15) compared to the basic setting. This shows us that iterated learning does have a clear effect on the structure of communication systems even in the Quantifier Game. This effect just doesn't result in a compositional language according to any of the standard metrics. I hypothesize that if the Quantifier Game required information about *all* attributes in the transformed object space, then iterated learning would encourage more compositional languages as measured by `topsim`, though quantifiers would still not emerge, just as number terms do not emerge in the Bag-Select Game.

This analysis of iterated learning's effect on the languages in the Bag-Select Game and the Quantifier Game begs the question of exactly what iterated learning is doing in other experiments. Does iterated learning always encourage the mapping of a single token to a single relevant *(attribute, value)* pair? And how exactly can we determine what the relevant attributes are? I have described elsewhere in this thesis a transformed object space that has two attributes, one for object type and one for sector. Why does iterated learning not encourage a mapping of one token to each *(attribute, value)* pair in that space, which would result in a non-trivially compositional language? It could be that the structure of the game itself tells the agents which attribute space is the most informative, and then iterated learning will minimize the number of tokens used to describe objects in that space.

Chapter 6
CONCLUSION AND FUTURE WORK

I presented a new game called the Quantifier Game, designed to study conditions that encourage the emergence of quantifiers in a communication system. I also introduced a method to measure the degree to which quantifiers have emerged in a system. This involves a new metric that uses differences in mutual information to measure how well a certain attribute is uniquely captured by a position or symbol in the message space. The metric for quantifier emergence also involves the transformation of the basic object space into the quantifier space, where there is one attribute that indicates object type, and another that indicates the quantity of that object type.

I find that quantifiers do not naturally emerge as a result of two agents playing this game. I find that compression and curriculum learning do not encourage quantifiers to emerge. However, I find that certain implementations of iterated learning do result in a higher quantifier score than the baseline.

Finally, my results point to the limitations of iterated learning in encouraging compositional languages as measured by `topsim`. An analysis of my results and the results of the Bag-Select Game of Guo et al. [8] suggest that iterated learning might be minimizing the number of symbols or tokens that are used to refer to relevant *(attribute, value)* pairs in some informative object space. This results in high `topsim` scores in some but not all cases.

6.1 Future Work

This thesis opens the door to much future work, including straightforward modifications to the experiments I presented and further exploration of some of the themes that I touch on.

As a direct extension, it would be interesting to try curriculum learning with a different

first stage. The first stage that I designed encouraged the agents to communicate about object types, and in some cases the final language after stage two did have a higher type score. Future work could include a first stage that emphasizes the importance of the sectors instead of the object types.

Another extension would be to explore the compression pressure and curriculum learning combined with iterated learning. Although curriculum learning didn't show many significant effects, there is a chance that combining it with iterated learning would change that. Combining compression and iterated learning seems like a more promising idea, though when I tried this later generations failed to converge on a successful protocol as message length completely collapsed. Future work could look at how to find the best value for the length lambda across generations.

Another interesting avenue to explore is the limits of iterated learning. Based on an analysis of the Quantifier Game and the Bag-Select Game, it is possible that iterated learning encourages a single token per relevant *(attribute, value)* pair, such as *0 of shape A* or *a lot of shape B*. To find non-trivial compositionality, it seems like attribute and value need to be disentangled, or that somehow the agents needs to recognize a more abstract attribute space as the relevant one. In the Quantifier Game this would mean that one token indicated object type, and another indicated sector. An interesting question is whether iterated learning will ever encourage that kind of disentanglement, or whether a different strategy is required to encourage the agents to convey information about a transformed object space.

A related question to explore is how to best implement iterated learning in any situation. This thesis highlighted the fact that there are many different ways to implement iterated learning, and these implementations can have different effects. It would be very valuable to have a roadmap for how to find the best iterated learning settings in any experiment.

I hope that future work can explore these questions and make further progress towards understanding the emergence of quantifiers.

BIBLIOGRAPHY

[1] Jacob Andreas. Measuring compositionality in representation learning. In *International Conference on Learning Representations*, 2019.

[2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[3] Henry Brighton and Simon Kirby. Understanding linguistic evolution by visualizing the emergence of topographic mappings. *Artificial life*, 12(2):229–242, 2006.

[4] Fausto Carcassi, Shane Steinert-Threlkeld, and Jakub Szymanik. The emergence of monotone quantifiers via iterated learning. In *Proceedings of the 41st annual meeting of the cognitive society (CogSci 2019)*, pages 190–196, 2019.

[5] Rahma Chaabouni, Eugene Kharitonov, Diane Bouchacourt, Emmanuel Dupoux, and Marco Baroni. Compositionality and generalization in emergent languages. In *ACL 2020-8th annual meeting of the Association for Computational Linguistics*, 2020.

[6] Morten H Christiansen and Simon Ed Kirby. *Language evolution.* Oxford University Press, 2003.

[7] Michael Cogswell, Jiasen Lu, Stefan Lee, Devi Parikh, and Dhruv Batra. Emergence of compositional language with deep generational transmission. *arXiv preprint arXiv:1904.09067*, 2019.

[8] Shangmin Guo, Yi Ren, Serhii Havrylov, Stella Frank, Ivan Titov, and Kenny Smith. The emergence of compositional languages for numeric concepts through iterated learning in neural agents. *EvoLang*, 2020.

[9] Sheng Guo, Weilin Huang, Haozhi Zhang, Chenfan Zhuang, Dengke Dong, Matthew R Scott, and Dinglong Huang. Curriculumnet: Weakly supervised learning from large-scale web images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 135–150, 2018.

[10] Serhii Havrylov and Ivan Titov. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. In *Advances in neural information processing systems*, pages 2149–2159, 2017.

[11] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparametrization with gumble-softmax. In *International Conference on Learning Representations (ICLR)*. OpenReview. net, 2017.

[12] Eugene Kharitonov, Rahma Chaabouni, Diane Bouchacourt, and Marco Baroni. EGG: a toolkit for research on Emergence of lanGuage in Games. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*. Association for Computational Linguistics, 2019.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[14] Simon Kirby and James R Hurford. The emergence of linguistic structure: An overview of the iterated learning model. In *Simulating the evolution of language*, pages 121–147. Springer, 2002.

[15] Simon Kirby, Monica Tamariz, Hannah Cornish, and Kenny Smith. Compression and communication in the cultural evolution of linguistic structure. *Cognition*, 141:87–102, 2015.

[16] Tomasz Korbak, Julian Zubek, and Joanna Rkaczaszek-Leonardi. Measuring non-trivial compositionality in emergent communication. *ArXiv*, abs/2010.15058, 2020.

[17] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 2873–2882. PMLR, 2018.

[18] Nur Geffen Lan, Emmanuel Chemla, and Shane Steinert-Threlkeld. On the spontaneous emergence of discrete and compositional signals. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4794–4800, 2020.

[19] Angeliki Lazaridou and Marco Baroni. Emergent multi-agent communication in the deep learning era. *arXiv preprint arXiv:2006.02419*, 2020.

[20] David Lewis. *Convention: A philosophical study*. John Wiley & Sons, 2008.

[21] Diana Rodríguez Luna, Edoardo Maria Ponti, Dieuwke Hupkes, and Elia Bruni. Internal and external pressures on language emergence: Least effort, object constancy and frequency. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4428–4437, 2020.

[22] Martin A. Nowak and David C. Krakauer. The evolution of language. *Proceedings of the National Academy of Sciences*, 96(14):8028–8033, 1999.

[23] Barbara Partee. Many quantifiers. In *Proceedings of ESCOL*, volume 5, pages 383–402, 1988.

[24] Yi Ren, Shangmin Guo, Matthieu Labeau, Shay B Cohen, and Simon Kirby. Compositional languages emerge in a neural iterated learning model. In *International Conference on Learning Representations (ICLR)*, 2019.

[25] Mathieu Rita, Rahma Chaabouni, and Emmanuel Dupoux. "lazimpa": Lazy and impatient neural agents learn to communicate efficiently. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 335–343, 2020.

[26] Brian Skyrms. *Signals: Evolution, learning, and information*. Oxford University Press, 2010.

[27] Shane Steinert-Threlkeld. Towards the emergence of non-trivial compositionality. *Philosophy of Science*, 2019.

[28] Zoltán Gendler Szabó. Compositionality. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2020 edition, 2020.

[29] Nicholas Tomlin and Ellie Pavlick. Incremental pragmatics and emergent communication. In *Neural Information Processing Systems Workshop on Emergent Communication*, 2018.

[30] Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, 2020.