CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Exploring the Evolution of Multi-Agent Emergent Languages through Neural Iterated Learning

A Computational Approach to Understanding Language Development and Complexity in AI Systems

Master's thesis in Computer Science and Engineering

Xiaolingzi Hu, Muyao Dai

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's Thesis 2024

Exploring the Evolution of Multi-Agent Emergent Languages through Neural Iterated Learning

A Computational Approach to Understanding Language Development and Complexity in AI Systems

Xiaolingzi Hu, Muyao Dai

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Exploring the Evolution of Multi-Agent Emergent Languages through Neural Iterated Learning
A Computational Approach to Understanding Language Development and Complexity in AI Systems
Xiaolingzi Hu, Muyao Dai

Supervisor: Jonathan Thomas, Emil Carlsson, Department of Data and Information Technology
Examiner: Devdatt Dubhashi, Department of Data and Information Technology

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Gothenburg, Sweden 2024

Exploring the Evolution of Multi-Agent Emergent Languages through Neural Iterated Learning
A Computational Approach to Understanding Language Development and Complexity in AI Systems

Xiaolingzi Hu, Muyao Dai
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

This study investigates the use of Neural Iterated Learning (NIL) in generating compositional languages within symbolic data systems and complex image datasets, such as MNIST. Through experiments focused on vocabulary size, message length, and penalization, we demonstrate that NIL can effectively produce structured communication systems, achieving high topological similarity and accuracy, particularly when using SoftMax for decision-making, Reinforce as the optimization method, and LSTM as the underlying network architecture. As the penalty introduced on message length, the increment of length cost leads to more efficient agent communication, with shorter and more structured messages, supporting the Brevity Law. While NIL performs robustly in simpler settings, its ability to maintain compositionality declines with increased data complexity, as observed in the Colored MNIST experiments. The results indicate that compressible representations improve generalization. However, NIL's performance in high-dimensional contexts is sensitive to input complexity, requiring refined feature extraction and training setups for improved stability and efficiency.

Keywords: Computer Science, Neural Iterated Learning, Natural Language Processing, Reinforcement Learning, Emergent languages, Language Evolution, Communication Efficiency, Compositionality, Feature Extraction, MNIST.

Acknowledgements

We would like to express the deepest gratitude to our advisors, Dr. Jonathan Thomas and Dr. Emil Carlsson, for their invaluable guidance, support, and encouragement throughout the course of this project. We are also profoundly thankful to our friends and families for their unwavering support throughout this entire journey.

Xiaolingzi Hu, Muyao Dai, Gothenburg, Monday 14th October, 2024

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The principle of *compositionality*, one of the most critical notions in the philosophy of language and logic, claims that natural languages express complex concepts through systematical assemblages of simple ones, each denoting elementary meanings (Partee, 2004). Modern semanticists interpret this claim following Montagovian tradition, which asserts that the whole meaning of a complex expression should be the function of the meanings of its atomic components and the syntactic mode of their combinations (Montague, 1973; Nefdt, 2020; Janssen and Zimmermann, 2021). With the preference for structure, human language semantics evolve to incarnate the nature of compositionality, the capability to convey an open-ended message set with a limited vocabulary (Ren et al., 2020). This principle has resulted in great interest in the field of artificial intelligence in recent years. Numerous studies have explored this phenomena and its application, with notable examples including the motivations for languages to be compositional in deep learning (Cogswell et al., 2019), the hierarchical models of the evolution of the compositional language (Barrett et al., 2018), and the use of grounded compositional language in multi-agent systems (Mordatch and Abbeel, 2018).

Cultural transmission has been suggested as a pressure to communicate efficiently, the repeated transmission of language from generation to generation, coupled with the pressure to communicate effectively, has promoted the emergence of combined language systems (Carr et al., 2020). To simulate this evolution process of language, *Iterative learning (IL)* is normally used, where structured mappings between signals and meanings emerged through one individual's behaviour learning by exposure to another's (Kirby et al., 2014). The integration of Iterative Learning Model(ILM) and neural networks has drawn great attention in recent decades, particularly in the field of evolutionary computation. Previous works have proven that iterated learning can produce systematically structured behaviour with the methods of computer simulation and mathematical modeling for both animal and non-animal languages (Griffiths and Kalish, 2007). Still, limitations such as information loss, simplicity convergence, and lack of generalization, have been faced in the application of iterative learning in natural language processing (Ren et al., 2023).

In response to these limits, Reinforcement Learning (RL) shows up as a strong method which is much more focused on goal-directed learning from interaction than any other approaches to machine learning. Agents in RL learn from its own experience - the trials, rewards, and errors to form the optimal strategies with explicit goals (Kirby et al., 2015). One of the major challenges that arise in RL is how to balance the exploration and the exploitation (Carlsson et al., 2023). However, the incorporation of RL into the model would allow researchers to investigate how the communication strategies between agents evolves within an uncertain environment (Sutton and Barto, 2018b).

Previous studies have investigated the application of iterated learning both in the lab with human participants and in artificial agents. Kirby et al. (2014) examined different approaches to explain how language evolves through the process of iterated learning, highlighting how this framework has been employed to account for the emergence of linguistic structure. Griffiths and Kalish (2007) demonstrated the use of Bayesian agents in iterated learning and how this model can reflect the inductive biases of learners.

1.1 Our Contribution

This research will explore the dynamics of communication and the principles of Neural Iterated Learning (NIL) within artificial neural networks. By examining how languages evolve and adapt in these artificial systems, particularly through the lens of NIL, we aim to provide a thorough understanding of the processes at play. In particular, we study neural networks that contain multiple agents engaged in a referential game that receives stimuli input from different datasets within a framework utilizing LSTM (Long Short-Term Memory) and Iterated Learning.

Building on the foundation of Ren Yi's model (Ren et al., 2020), we designed and conducted a series of experiments to answer the following questions: (1) How does the training outcomes differ for our datasets with Reinforcement Learning and Neural Iterative Learning? (2) How did the type of input data, either textual or pixel-based, influence the linguistic structure of the emergent language? (3) How does the message features such as message length and length of added vocabulary affect the compositionality of the emergent language? (4) When constraints are put onto the properties of the message, such as message length, how would it influence the network's performance?

The remainder of this paper is constructed as follows: First, we provide a background review on Neural Iterative Learning(NIL) and pivotal linguistic concepts from a cognitive science perspective. Then we detail our experiments, including the multi-agent model implemented with Reinforce and NIL on both textual and graphic datasets, with a comprehensive introduction on the corresponding methodology. Extended trials with respect to message features are also explicitly explained. Then we present the analysis methods we adopted and the results we obtained. Based on the results, we discuss the implications for further studies.

Chapter 2

Background

2.1 Compositionality

As introduced previously, compositionality is one of the fundamental design features in human languages (Hockett, 1960). A growing amount of literature has demonstrated the remarkable potential of deep networks which simulate the emergence of compositionality in human language (Kirby et al., 2008). Pressure out of the need for communication and transmission brings about this emergence of language structure. Previous studies have corroborated that language derives meanings from its usage Wagner et al. (2003) and more structured language with higher compositionality is likely to show up when the agents perceives the world more structured (Lazaridou et al., 2018). Smaller units such as morphemes or words are combined to create an unbounded variety of complex structures. Therefore, the meaning of these larger structures is derived from that of the individual components and the way they are composed (Frege, 1892).

This compositional nature has drawn significant attention from researchers in artificial intelligence as it highlights the potential that, infinity of expressions could be built up through a finite dictionary and a finite set of construction rules. The recursive characteristic in language construction ensures efficiency and conciseness for the neural networks, especially where the computational ability or memory space is limited (Kirby, 2002). With the scalability enabled by the recursive combination, it is possible to handle the growing complexity of tasks and environments.

2.2 Iterative Learning

Iterated learning has been applied to explore a variety of properties, including compositionality, regularization(Kirby et al., 2015), and generalization (Hupkes et al., 2020), which are crucial for building an efficient semantic system. To accomplish different interactive tasks, such as emergent communication tasks or visual question answering, IL is introduced as an effective artificial method in deep learning models

(Vani et al., 2021a). IL in deep learning has been proved to advantageous in improving the degree of compositionality of the neural representation in deep models (Ren et al., 2020), enhancing the compositional generalization in general representation learning (Ren et al., 2023) and amplifying the hidden bias within the intelligent agents (Ren et al., 2024).

Languages are transmitted across generations among humans and throughout generations by means of iterated learning: people learn a language from the ones who have learnt the language themselves (Griffiths and Kalish, 2007). Some researchers see languages as independent, complex, and evolving, that is shaped by selection pressures from both human learning and the processing mechanism. Each generation of language users imposes these pressures, suggesting that, those seemingly arbitrary linguistic structures arise from general biases in cognition, perception, motor functions, and pragmatics (Christiansen and Chater, 2008). Throughout this process, features evolve to become more learnable, structured and adaptive linguistic structures emerge cumulatively in response to the challenges of cultural transmission, even in the absence of deliberate intent (Kirby et al., 2008).

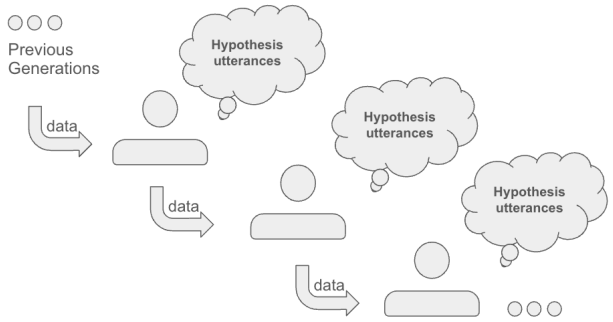


Figure 2.1: Iterative Learning in Language Transmission: each learner generates utterances with hypothesis based on the languages data provided by the previous generation

As illustrated in Figure 2.1, the mechanism of peer-to-peer learning in IL, accelerates the emergence of combinatorial structures (Kirby and Tamariz, 2022). When this repeated transmission is combined with communicative pressures, compositional language begins to evolve (Kirby and Tamariz, 2022), towards the optimal balance of expressivity and simplicity (Kirby et al., 2015).

2.3 Neural Iterated learning

Building on the foundational concepts of neural networks and cognitive modeling, the integration of neural agents into the referential game framework, coupled with an iterative learning model that mirrors human knowledge transmission, allows agents to effectively simulate the language evolution process within a context that closely

resembles human cognitive science (Lazaridou et al., 2017).

Guo et al. (2019) and Ren et al. (2020) explored the emergence of compositional languages for numeric concepts through iterated learning in neural agents. They demonstrated that structured languages can evolve through neural iterated learning, providing learning speed advantages to neural agents during training. Kirby et al. (2015) and Vani et al. (2021b) further investigated the role of iterated learning in the emergence of compositional languages in machine learning tasks, emphasizing the need for explicit pressure to drive the development of efficient, systematic further compositional languages.

The previous study in this field found that neural agents are given a learning speed due to the high compositional language. In our experiment, this is readily validated using textual inputs structured as adjective-noun pairs, which we can directly convert into one-hot encoded formats. Conversely, when dealing with pixel-based inputs, we must first employ an image agent to translate the pixel information into embeddings. Direct validation of the compositionality of these embeddings is challenging due to their transformation through fully connected layers using ReLU activations. As a result, the pre-linguistic object becomes entangled within the image agent.

2.4 Reinforcement Learning

Reinforcement Learning (RL) focuses on enabling an artificial agent to learn optimal actions in an unknown environment. If the agent only exploits known strategies, it may miss better rewards in unexplored areas. Conversely, solely exploring prevents the agent from leveraging the knowledge it has gained. Therefore, achieving an optimal balance between exploitation and exploration is essential when dealing with sequential decision problems (Sutton and Barto, 2018a). Deep neural networks have prevailed in RL in recent years, especially in areas of robotics, gaming (Mnih et al., 2015), autonomous vehicles (Levine et al., 2015; Lillicrap et al., 2019), natural language processing (NLP) (Zaremba et al., 2016), etc. In this study, we will integrate RL as part of the standard NIL setup, rather than making it the primary focus, as it plays a key role in facilitating NIL's iterative learning process.

2.4.1 LSTM

Long Short-Term Memory (LSTM), a type of recurrent neural network (RNN), excels in modeling temporal sequences and capturing long-term dependencies, addressing limitations of traditional RNNs that struggle with retaining information over extended time steps. LSTM introduces a "forget gate" that adaptively resets its internal state, preventing indefinite state growth and ensuring stable learning over continuous input streams, which traditional RNNs fail to manage effectively (Gers et al., 2000; Sak et al., 2014). Figure 2.2 illustrates a single memory block of the LSTM architecture.

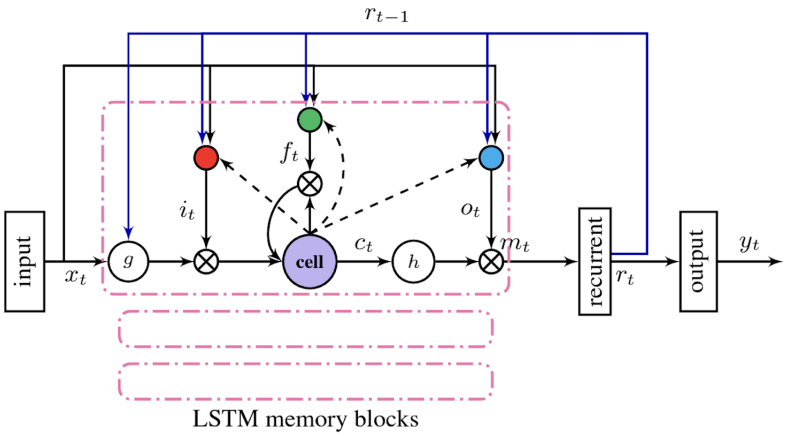


Figure 2.2: LSTM architecture (a single memory block) (Peralta et al., 2022)

2.5 Zipf's Law of Abbreviation

In human language, George Kingsley Zipf discovered that word frequency tends to be inversely correlated with its length, a phenomenon known as Zipf's Law of Abbreviation (ZLA) (Zipf, 1949). It has been observed that languages evolve towards shorter representation strings—essentially effort minimization—under functional pressures, such as cultural transmission and communication demands (Kirby et al., 2008). Zipf also found that human word distribution is highly skewed, from the perspective of lexical statistics, and recent studies confirm that random-typing distributions also follow this law. However, ZLA does not appear to hold in the context of computational language within a Reinforce multi-agent model. Unlike human language, where abbreviating for "articulatory" effort is beneficial, artificial communication does not require this reduction (Zipf, 1949). The neural network seems to exhibit a bias on the listener side, where longer messages incur no extra cost but provide more information, thus enhancing task performance, similar to artificial human communication tasks (Kanwal et al., 2017). In fact, recent research suggests that the emergent communication system developed by neural networks tends to be anti-efficient, favoring longer messages for more frequent inputs, which is contrary to ZLA. This pattern likely stems from the asymmetry between speaker and listener biases (Chaabouni et al., 2019). In this study, we will investigate whether the ZLA holds true in the NIL model, exploring if the same principles apply.

2.6 Referential Game

The referential game, developed within the framework of the signaling game introduced by Lewis (1969), is extensively used in economic and game theory to model communication between individuals (Spence, 1973; Cho and Kreps, 1987). Unlike traditional signaling games, referential games are specifically designed to explore how linguistic communication can emerge and develop, by involving agents—a

sender and a receiver—who must coordinate to identify objects or concepts through communication.

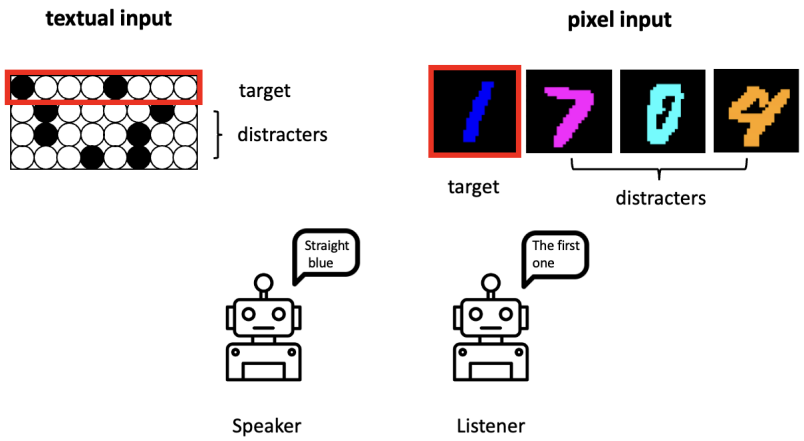


Figure 2.3: Illustration of the referential game

In our game, agents are designated as speakers and listeners, as shown in figure. The objects are denoted by the set $O = \{o_1, \ldots, o_N\}$. From this set, a target $t \in O$ is selected, along with $K-1$ distractors $D = \{d_1, \ldots, d_{K-1}\} \subset O$ such that $\forall j, t \neq d_j$. Each round, the speaker is informed of the target t and generates a discrete message as their lexicon to send to the listener, who is presented with the candidate set $C = \{t\} \cup D$, without knowing which element is the target. To promote the development of semantically meaningful language, the speaker is rewarded if the listener correctly identifies the target t among the candidates based on the message. To prevent recognition of objects by their presentation order, both the objects and candidates are randomly shuffled each round.

2.7 Topological Similarity

Building upon the multi-word messages dataset, we can explore more complex feature of language structure, particularly compositionality. While there is no exhaustive method to measure compositionality, Topological similarity offers a quantitative approach, as discussed by Brighton and Kirby (2006).

Consider a language function $L(\cdot) : X \to M$ that maps objects in space X to multi-word messages in space M. To measure similarity, distances between object pairs in X are calculated as $\Delta_{ij} = d_X(x_i, x_j)$, where $d_X(\cdot)$ is the distance metric in object space. Correspondingly, distances in the message space M are computed as $\Delta_{ij} = d_M(m_i, m_j)$, with $m_i = L(x_i)$ and $d_M(\cdot)$ as the message space distance metric. Topological similarity(denoted as ρ) is then determined by the correlation of these distances across X.

There is substantial research within the field of neural iterated learning at the intersection of linguistic science. As stated in the background section, compositionality is considered a critical measurement of emergent language. Complex language structures can be understood clearly in natural language by decoding language into smaller pieces, such as words and phrases.

In previous experiments, Ren et al. (2020) demonstrated that high topological similarity offers advantages to both listening and speaking agents in neural iterated learning algorithms. However, within this algorithmic framework, it remains unclear which factors contribute most significantly to the emergence of these perfect languages. To investigate which aspects play a crucial role during the interaction and learning phases and how knowledge from the previous phase aids in the next generation, we first reproduced Ren Yi's topological similarity experiment using a simple text phrase dataset. Following this, we conducted experiments focusing on the structure of the messages, with particular attention to message length and vocabulary size, as these elements are likely to be closely linked to the development of emergent languages.

2.8 Hypothesis

We set out our predictions on the performance of compositionality which can be tested in our later experiments as follows:

1. Neural iterated learning, which inherently includes a communication step, will effectively forming the language with high compositionality and expression. And the inclusion of LSTM would improve the model's overall performance.

2. Under the framework of Neural iterated learning, we hypothesize that the compressibility of representations is positively correlated with generalization, as the underlying structure of the data is captured by more compacted representations, the model's performance is thus enhanced on the new tasks.

3. NIL is able to exhibit consistent success in producing highly compositional language across both symbolic and image datasets. However, as the complexity of the image data increases, NIL's ability to develop compositional language is expected to diminish.

4. The communication of the network will not develop towards brevity unless an artificial penalty is imposed, aligning with the principles of Zipf's Law of Abbreviation (ZLA). As the penalty increases, the messages will converge into more concise and efficient forms.

5. In the absence of this penalty, the neural network may exhibit an anti-efficient communication pattern, favoring longer messages that provide more information for the listener network without considering expression effort minimization.

Chapter 3

Model

This chapter presents the neural iterated learning model, focusing on the key components: agents, the algorithm, the loss function, and the evaluation metrics.

We begin by introducing the agents in the model, specifically the speaker and listener. The speaker encodes the target object into a message, while the listener decodes this message to identify the target. These agents interact over multiple generations, learning to communicate through iterative processes.

The loss function used in this model is then introduced, which measures the difference between the predicted and true outputs. By minimizing this loss, the model adjusts the agents' behavior to improve the accuracy of their communication.

Next, the algorithm driving their interaction is explained. The pseudocode outlines how the speaker and listener exchange information, with each generation adjusting based on the knowledge received from the previous one.

Finally, the chapter covers the evaluation methods used to assess the model's performance. Speaker accuracy and topological similarity are the primary metrics, helping to evaluate both the effectiveness and structure of the emergent communication system.

3.1 Agents

3.1.1 Speaker Agent

The Speaker agent is responsible for encoding the target t, which can be either symbolic or pixel-based, into a message m that is communicated to the Listener agent. The input t is processed by an encoder $f(t; \theta_f)$ to produce a dense feature representation u. The encoder is implemented as a multi-layer perceptron (MLP) with three hidden layers, each followed by a ReLU activation function, as shown

below:

$$u = f(t; \theta_f) = \text{ReLU}(\mathbf{W}_3 \cdot \text{ReLU}(\mathbf{W}_2 \cdot \text{ReLU}(\mathbf{W}_1 \cdot t) + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3)$$

where $\mathbf{W}_i$ and $\mathbf{b}_i$ are the weights and biases of the i-th layer of the MLP, respectively.

Given the encoded representation u, the speaker generates a discrete message m by sequentially sampling symbols from a fixed vocabulary $\mathcal{A}$, which has a predefined size V. This vocabulary represents a limited set of symbols that the model can choose from at each time step, where V is a constant value specified during the initialization of the model.

The recurrent policy $\pi_{\mathcal{S}}$ is implemented by an LSTM-based decoder $g_{\mathcal{S}}(u; \theta_{g_{\mathcal{S}}})$, which generates a sequence of symbols from the vocabulary. At each time step, the LSTM computes a hidden state h_t and cell state c_t, which are used to produce a logit vector over the vocabulary. The logits are then converted into probabilities using a softmax function, allowing the model to sample the next symbol from the vocabulary.

Formally, the message generation process can be described as follows:

$$\begin{aligned} h_0, c_0 &= u, 0 \\ h_t, c_t &= \text{LSTMCell}(x_t, (h_{t-1}, c_{t-1})) \quad \forall t \in \{1, \ldots, L\} \\ \text{logits}_t &= \mathbf{W}_{\text{out}} \cdot h_t + \mathbf{b}_{\text{out}} \\ \pi_{\mathcal{S}}(a_t | h_t) &= \text{Softmax}(\text{logits}_t) \end{aligned}$$

where:

- L is the maximum message length.
- h_t and c_t are the hidden and cell states of the LSTM at time step t.
- logits_t are the unnormalized log probabilities over the fixed vocabulary $\mathcal{A}$, which has a size V.
- $\pi_{\mathcal{S}}(a_t | h_t)$ is the probability distribution over the vocabulary at time step t, obtained via the softmax function applied to the logits.

During training, symbols are sampled from the distribution $\pi_{\mathcal{S}}$, while during inference, the model selects the symbol with the highest probability at each time step.

3.1.2 Listener Agent

The Listener agent processes the received message m to identify the target object among a set of candidate objects $X_{\text{candidates}}$. Each candidate $x \in X_{\text{candidates}}$ is

encoded into a dense representation u using the same encoder $f(t;\theta_f)$. The encoded message m is then processed by an LSTM decoder $h_{\mathcal{L}}(m;\theta_{h_{\mathcal{L}}})$ to obtain a message representation z:

$$z = h_{\mathcal{L}}(m;\theta_{h_{\mathcal{L}}})$$

The Listener's task is to predict the target object $\hat{t}$ by selecting the candidate x whose representation u_x has the highest similarity to z. This is achieved by computing the dot product between z and each candidate's encoded representation u_x and applying a softmax function to obtain a probability distribution over the candidates:

$$\pi_{\mathcal{L}}(\hat{t}|z,t) = \text{Softmax}\left(\{\langle z, u_x\rangle : x \in X_{\text{target}}\}\right)$$

During training, the listener agent samples a candidate object according to this distribution, while at inference time, the object with the highest probability is selected greedily.

3.2 Loss function

The training process for the speaker and listener agents involves optimizing their respective loss functions to enhance the communication success between the two agents. The objective is to ensure that the message generated by the speaker enables the listener to correctly identify the target object.

3.2.1 Speaker Loss

The speaker's loss function depends on the method used for generating messages. In the mostly used reinforce setting, the speaker's loss is determined by the difference between the expected reward and the entropy of the speaker's message policy. This approach allows the speaker to explore different strategies for message generation while being guided by the feedback from the listener's performance.

The speaker's loss function can be written as:

$$\mathcal{L}_{\mathcal{S}} = -\mathbb{E}_{\pi_{\mathcal{S}}}\left[R(\hat{t},t)\log\pi_{\mathcal{S}}(m|u,x_{\text{target}})\right] + \lambda H(\pi_{\mathcal{S}})$$

where:

- $R(\hat{t},t)$ is the reward function.
- $\log\pi_{\mathcal{S}}(m|u,x_{\text{target}})$ is the log-probability of the message generated by the speaker given the message encoder u and the target x_{target}.

- $H(\pi_{\mathcal{S}})$ is the entropy of the speaker's policy, with λ controlling the strength of the regularization.

In the alternative, training modes such as Self-Critical Sequence Training (SCST) (Rennie et al., 2017) are implemented to compare to the reinforcement learning mode, where the speaker's loss is adjusted by comparing the rewards of the current policy to a baseline policy, thereby reducing variance in the gradient estimates and stabilizing training.

3.2.2 Listener Loss

The listener's loss measures how well the listener can decode the message from the speaker to correctly identify the target object from a set of candidates. To further refine the listener's decision-making process, an entropy regularization term can be introduced, which promotes exploration by preventing the listener from becoming overly confident in its predictions.

The listener's loss function can be written as:

$$\mathcal{L}_{\mathcal{L}} = -\mathbb{E}_{\pi_{\mathcal{L}}} \left[R(\hat{t}, t) \log \pi_{\mathcal{L}}(t|z, x_{\text{candidates}}) \right] + \alpha H(\pi_{\mathcal{L}})$$

where:

- $\log \pi_{\mathcal{L}}(t|z, x_{\text{candidates}})$ is the log-probability assigned by the listener to the correct target t, given the message encoding z and the candidates $x_{\text{candidates}}$.

- $H(\pi_{\mathcal{L}})$ is the entropy of the listener's policy, encouraging exploration by preventing overconfidence, with α controlling the regularization strength.

3.3 Algorithm

The model we implemented, based on the structure proposed by Ren et al. (2020), is structured with three major phases: Learning Phase, Interaction Phase, and Transmission Phase, as shown in Algorithm 1, the notation used is explained in Table 3.1.

Table 3.1: Notations used in Neural Iterative Learning

Notation	Description
$(X_{\text{target}}, X_{\text{candidates}} = \{X_{\text{target}}, X_{\text{distracter}_1}, X_{\text{distracter}_2}, \ldots, X_{\text{distracter}_n}\} \leftarrow \mathcal{D}$	Dataset
$(X_{\text{target}}, \mathcal{M}_{\text{target}}) \leftarrow \mathcal{K}$	Knowledge pair
$\mathcal{S}$	Speaker model
$\mathcal{L}$	Listener model
α	Learning accuracy
ρ	Topological similarity
$\mathcal{R}$	Reward function
L	Cross-entropy loss function
$\hat{t}, t$	Predicted label, True label
E_i	Number of epochs in interaction phase
E_l	Number of epochs for listener pre-training
E_s	Number of epochs for speaker pre-training

Algorithm 1 Training Algorithm

Learning Phase:
Input: Initial speaker $\mathcal{S}$, listener $\mathcal{L}$, knowledge pair $\mathcal{K}$ and dataset $\mathcal{D}$
for each epoch $e = 1, \ldots, E_s$ **do**
    Load inputs $(X, \mathcal{M}) \leftarrow \mathcal{K}$
    Speaker $\mathcal{S}$ generates predictions: $\mathcal{M} \leftarrow \arg\max(\mathcal{S}(X))$
    Update speaker $\mathcal{S}$ by minimizing $L \leftarrow \text{CrossEntropy}(\mathcal{M}, \hat{\mathcal{M}})$
end for
Compute Learning Accuracy $\alpha \leftarrow \frac{1}{|X|} \sum_{i=1}^{|X|} \mathbb{1}[\hat{\mathcal{M}}_i = \mathcal{M}_i]$
for each epoch $e = 1, \ldots, E_l$ **do**
    Load inputs $X \leftarrow \mathcal{D}$
    Speaker $\mathcal{S}$ generates message $m \leftarrow \mathcal{S}(X_{\text{target}})$
    Listener $\mathcal{L}$ receives m and predicts target $\hat{t} \leftarrow \mathcal{L}(m, X_{\text{candidates}}))$
    Compute reward $\mathcal{R} \leftarrow \frac{1}{|X|} \sum_{i=1}^{|X|} \mathbb{1}[\hat{t}_i = t_i]$
    Update listener $\mathcal{L}$ by minimizing listener loss using backpropagation
end for

Interaction Phase:
for each epoch $e = 1, \ldots, E_i - E_l$ **do**
    Load inputs $X \leftarrow \mathcal{D}$
    Speaker $\mathcal{S}$ generates message $m \leftarrow \mathcal{S}(X_{\text{target}})$
    Listener $\mathcal{L}$ receives m and predicts target $\hat{t} \leftarrow \mathcal{L}(m, X(\text{candidates}))$
    Compute reward $\mathcal{R} \leftarrow \frac{1}{|X|} \sum_{i=1}^{|X|} \mathbb{1}[\hat{t}_i = t_i]$
    Update both the speaker $\mathcal{S}$ and the listener $\mathcal{L}$ by minimizing listener loss and speaker loss
end for

24: Compute Topological Similarity ρ over $\mathcal{M}$

25: **Transmission Phase:**
26: **for** $i = 1$ to I **do**
27: Generate a batch by randomly selecting one data point from each label in $\mathcal{D}_g$
28: Train the speaker $\mathcal{S}_g$ and listener $\mathcal{L}_g$ on this batch through reward
29: Append the resulting data to $\mathcal{D}_g^{(i)}$
30: **end for**
31: Combine all batches into a single dataset: $\mathcal{D}_g^{\text{combined}} = \bigcup_{i=1}^{I} \mathcal{D}_g^{(i)}$
32: Shuffle and reduce the combined dataset $\mathcal{D}_g^{\text{combined}}$ to generate a fixed-size dataset: $\mathcal{D}_{g+1} \leftarrow \text{Select}_P(\mathcal{D}_g^{\text{combined}})$
33: **Output:** New dataset $\mathcal{D}_{g+1}$

As mentioned in section 2.6, the referential game incorporates multiple settings to encourage the emergence of language and promote higher topological similarity. Drawing inspiration from human cognitive systems, the speaker acts as a teacher, transmitting information, while the listener learns by interpreting the message. However, unlike a typical student-teacher relationship, the listener's learning is guided by task-specific reinforcement signals rather than direct instruction, with the transmission phase involving a reward-based mechanism to refine its understanding.

3.3.1 Learning Phase

In this phase, the speaker and listener learn through distinct processes. The speaker learns independently by minimizing cross-entropy loss, refining its communication strategies based on knowledge from the previous generation without interaction with the listener. This allows the speaker to focus solely on reproducing messages accurately. In contrast, the listener learns through reinforcement learning (RL) loss during interaction with the speaker. The listener's learning depends on feedback from the communication process, allowing it to refine its ability to decode the speaker's messages into meaningful knowledge.

In the context of our referential game, the learning of the listener is motivated by cultural evolution frameworks, as discussed by Kirby and Tamariz (2022). Pre-training allows the listener to learn and adapt to the speaker's messages early on, providing a more stable foundation for communication and reducing the noise and instability that can arise when agents start from scratch. This is particularly important given that the first generation needs to gain prior knowledge.

The speaker is initialized for the new generation, and inputs $(X, \mathcal{M})$, representing objects and their corresponding messages, are loaded from the knowledge pair $\mathcal{K}$ (line 4). The speaker generates predictions $\hat{\mathcal{M}}$ based on these inputs (line 5), where $\hat{\mathcal{M}} = \arg\max(\mathcal{S}(X))$. To evaluate the speaker's performance, cross-entropy loss L is computed between the predicted messages $\hat{\mathcal{M}}$ and the expected messages $\mathcal{M}$ (line 6). The learning accuracy α is then determined as the proportion of correct predictions (line 8), reflecting how effectively the speaker has learned from the

previous generation.

In the listener's learning phase, only the listener's parameters are updated. Inputs X are loaded from the dataset $\mathcal{D}$, and the speaker generates a message m based on the target object X_{target} (line 10-11). The listener receives the message m and predicts the target $\hat{t}$ from the set of candidates $X_{\text{candidates}}$ (line 12). The reward $\mathcal{R}$ is computed as the proportion of correct predictions made by the listener (line 13), and the listener is updated by minimizing its loss function (line 14). Only the listener is updated at this stage, allowing it to adapt to the speaker's messages.

3.3.2 Interaction Phase

After the listener has been trained, the system moves into the interaction phase. Here, both the speaker and listener are updated together, refining their communication strategies.

The interaction phase continues for $E_i - E_l$ epochs. During this phase, the speaker generates messages from input objects, and the listener attempts to interpret these messages by predicting the target (lines 18-20). The reward $\mathcal{R}$ is calculated as the proportion of correct target predictions made by the listener (line 21), and both the speaker and listener are updated based on their respective loss functions (line 22). This iteraction process enhances the communication system by refining both the generation and interpretation of messages.

Finally, the system computes the topological similarity ρ over the set of generated messages M (line 24). Following the interaction phase, the system enters the transmission phase.

3.3.3 Transmission Phase

In the transmission phase, new data is generated in the form of knowledge pairs, where each pair consists of an object and its corresponding message, represented as $(X, M) \in \mathcal{D}_{g+1}$.

To obtain $\mathcal{D}_{g+1}$, the current speaker $\mathcal{S}_g$ utilizes $\mathcal{D}_g$ to produce these knowledge pairs (X, M), which reflect the speaker's current understanding of the objects. This process begins by generating multiple batches, each containing one randomly selected data point from each label in $\mathcal{D}_g$ (line 27). The speaker and listener are trained on each batch, resulting in corresponding datasets $\mathcal{D}_g^{(i)}$ (line 28).

These datasets are then combined into a single dataset (line 29). To prevent overfitting and enhance generalization, the combined dataset is shuffled and reduced in size by selecting a fixed number of data points, P, to create the final dataset for the next generation (line 32). The resulting refined dataset, $\mathcal{D}_{g+1}$, is then prepared for use in the subsequent phase.

3.4 Evaluation

Three evaluation matrices are adopted: the reward obtained in the referential game, topological similarity, and speaker accuracy. The reward measurement is straightforward, based on the agent's success rate in correctly identifying the target during the game. In contrast, assessing topological similarity requires a more careful approach, as it does not rely on direct feedback and involves evaluating structural patterns in the emergent language. Meanwhile, speaker accuracy measures the proportion of correct reproductions of the previous generation's messages.

3.4.1 Reward

The reward function, which returns 1 if the listener correctly identifies the target ($\hat{t} = t$) and 0 otherwise. The reward function is structured to account for correct predictions, considering direct matches and instances where the predicted label is equivalent to the selected label, particularly in cases where the candidate set is large and diverse.

$$\mathcal{R}(\hat{t}, t) = \mathbb{1}[\hat{t} = t]$$

where:

- t is the true target label.
- $\hat{t}$ is the label predicted by the listener.
- $\mathbb{1}[\hat{t} = t]$ is an indicator function that returns 1 if $\hat{t} = t$ (i.e., the listener correctly identifies the target) and 0 otherwise.

3.4.2 Topological Similarity

We assessed the distance of object pairs $\Delta_{ij} = d_X(x_i, x_j)$ using hamming distance, named after Hamming (1950). Formally defined as the distance between two equal-length strings of symbols is the number of positions at which the corresponding symbols are different (Waggener, 1995). Although our dataset includes both symbolic and pixel data, we ensured consistency in the object structure across data types. For example, in the symbolic dataset, an object might be represented as a phrase like 'nice chair', while in the pixel dataset, it could be a visual input such as 'blue 8'. In both cases, the underlying structure remains consistent, allowing us to apply Hamming distance as a valid metric to compare these objects, as both types can be encoded into comparable symbolic forms, facilitating the evaluation of differences between them.

Measuring the distance between pairs of messages, denoted as $\Delta_{ij} = d_M(m_i, m_j)$, presents a more intricate challenge. In our experiment, each message m_i is represented

as a matrix of dimensions $[A, V]$, where A corresponds to the message length and V represents the vocabulary size per message. Notably, V is consistent across all messages. To quantify the dissimilarity between these message matrices, we employ the Levenshtein distance (Levenshtein, 1966), which calculates the minimum number of operations (insertions, deletions, or substitutions) required to transform one message into another.

The overall topological similarity was then assessed by analyzing the correlation between the concept distances (Hamming distance) and message distances (Levenshtein distance). A high correlation would indicate that similar concepts tend to produce similar messages, which is a desirable characteristic in a structured communication system.

3.4.3 Speaker Accuracy

Speaker Accuracy measures how well the speaker in the new generation reproduces the messages generated by the previous generation. The speaker's task is to generate a message $\hat{M}$ based on the input X, which is then compared to the expected message M from the previous generation. The accuracy is calculated as the proportion of correct predictions, where a correct prediction occurs if the generated message is identical to the expected message.

$$\alpha = \frac{1}{|X|} \sum_{i=1}^{|X|} \mathbb{1}[\hat{M}_i = M_i]$$

where:

- M_i is the expected message from the previous generation.
- $\hat{M}_i$ is the message predicted by the speaker in the new generation.
- $\mathbb{1}[\hat{M}_i = M_i]$ is an indicator function that returns 1 if $\hat{M}_i = M_i$ (i.e., the speaker correctly reproduces the previous generation's message) and 0 otherwise.

This metric provides insight into how effectively the speaker has learned from the previous generation, with higher accuracy indicating better retention and reproduction of the intended communication strategy. However, perfect retention is not always the desired outcome. In some cases, overly perfect reproduction of the previous generation's messages can inhibit the development of the emergent language in next generation. This balance between retention and innovation is explored further in the subsection 4.2.4, where we analyze how different levels of retention impact language development.

3.4.4 Evaluation across Input Types

The evaluation was conducted separately for symbolic and pixel-based inputs, recognizing that these different types of data might exhibit different patterns of Topological similarity and message generation. By comparing the results across these input types, we aimed to gain insights into the robustness of the agents communication system in handling diverse forms of input.

Symbolic Data

For symbolic inputs, we manually selected distinct attributes such as color and shape, which were combined to form phrases. These phrases were then converted into one-hot encoded vectors, where each unique phrase corresponds to a specific language description. The structured nature of symbolic data ensures that each concept is represented by a unique and precise combination of attributes, making the calculation of distances straightforward.

Pixel Data

In contrast, pixel-based inputs present a more complex scenario. Each label, such as "cat" or "dog" may correspond to various images, resulting in multiple objects being associated with the same concept. This inherent variability leads to challenges in calculating dissimilarities. Unlike symbolic data, where each concept has a precise representation, pixel data introduces ambiguity due to the diversity of visual representations for a single label. As a result, calculating distances between pixel-based concepts requires careful consideration of this variability.

To address this challenge, our approach involved randomly selecting a message from the set of all possible images corresponding to each label across several epochs and then averaging the results. For each epoch, we randomly chose one message for each label and then calculated the pairwise distances between all possible pairs of messages and concepts. This process involved:

1. For each epoch e, randomly select a message $m_i^{(e)}$ from $\mathcal{M}(l_i)$ for each label l_i. This yields a set of selected messages $M^{(e)} = \{m_1^{(e)}, m_2^{(e)}, \ldots, m_n^{(e)}\}$.

2. Form all possible pairs of concepts (x_i, x_j) and their corresponding messages $(m_i^{(e)}, m_j^{(e)})$ for each epoch.

3. Compute the Hamming distance $d_H(x_i, x_j)$ and the Levenshtein distance $d_L(m_i^{(e)}, m_j^{(e)})$.

4. Repeat the above steps across several epochs to capture the variability in message generation, and then average the results to obtain the final estimates:

$$\overline{d_H} = \frac{1}{E}\sum_{e=1}^{E} d_H^{(e)}, \quad \overline{d_L} = \frac{1}{E}\sum_{e=1}^{E} d_L^{(e)},$$

where E is the total number of epochs.

By averaging the results across these epochs, we aimed to mitigate the variability in the pixel data and obtain a more robust estimate of the Topological similarity. This method allowed us to better account for the inherent differences in visual representations while still applying consistent evaluation metrics across both symbolic and pixel-based data.

Chapter 4

Symbolic Experiment

In this section, we explore the impact of vocabulary length and message length on NIL within the domain of symbolic data. We conduct an empirical evaluation to investigate how these parameters influence the structure of the resultant emergent languages. Specifically, we assess these languages in terms of topological similarity and communication accuracy, key metrics that reflect the compositionality and effectiveness of the communication system. This demonstrates the significance of the model's structural decisions on language formation.

We further explore the impact of various training and configuration choices on this outcome. For instance, we assess the performance of methods such as REINFORCE, Self-Critical Sequence Training (SCST), and Gumbel-Softmax (GS), along with differing network architectures like GRU and LSTM, to determine how these variations influence the emergent language's compositionality and communication efficiency. We also look into how the configurations of the model and the message influence the final language formation.

Finally, noting that the selection of the message length is a hyperparameter, we attempt to automatically learn the optimal message length by utilizing a reward-based penalty. We find that introducing this penalty encourages more concise communication, leading to a balance between expressivity and efficiency, with shorter messages still retaining high levels of topological similarity and communicative success, thus testing whether Zip's Law of Abbreviation holds.

4.1 Dataset

In this section, we explain how the combinations of colors and objects in the dataset are represented through encoding techniques. The goal is to ensure that each element in these combinations is distinctly captured, providing a structured language format for the agent to process. The dataset is designed to abstractly represent the conceptual distinctions between colors and objects without attaching any inherent

real-world meaning.

The dataset $\mathcal{D}$ was constructed by pairing elements from two sets: a set of n distinct colors C and a set of n distinct objects O. For example, these sets might be defined as:

$$C = \{\text{Color}_1, \text{Color}_2, \ldots, \text{Color}_n\}$$

$$O = \{\text{Object}_1, \text{Object}_2, \ldots, \text{Object}_n\}$$

The dataset $\mathcal{D}$ comprises all possible combinations of elements from C and O, resulting in n^2 unique entries:

$$D = C \times O$$

For each combination $(c, o) \in D$, separate one-hot encoded vectors are generated for both the color c and the object o. The one-hot vectors for both color and object are of the same size n, chosen for the convenience of encoding these vectors into a unified message format. These vectors are concatenated into a single vector $\mathbf{v} \in \mathbb{R}^{2n}$, where the positions corresponding to the selected color and object are set to 1, while all other positions are set to 0. This encoding ensures that both the color and object identities are distinctly represented, allowing the model to process these combinations in a consistent and structured manner.

4.2 Baseline experiment

In this experiment, we investigated how Neural Iterated Learning (NIL) handles structured symbolic data in the context of a referential game. We reproduced the topological similarity experiment from Ren et al. (2020) with some adjustments following the algorithm outlined in 1. The primary goal was to study the evolution of the language used by the model during the communication process and to assess its ability to develop a highly compositional structure.

4.2.1 Experiment Setup

The experiment setup, as described in section 3.3, involves a speaker (subsection 3.1.1) who generates a message based on the symbolic input, and a listener (subsection 3.1.2), who aims to identify the target from several candidates based on the speaker's message. The controlled symbolic dataset, consisting of combinations of basic colors and objects, serves as a baseline for evaluating the model's performance through two key metrics: the reward obtained in the referential game, which indicates successful communication, and the topological similarity of the emergent language across iterations.

4.2.2 Controls and Variables

We aim to evaluate Neural Iterated Learning's ability to develop a structured communication system by measuring both the topological similarity and the reward obtained during the referential game. The baseline experiment focuses on two key aspects: whether the model can produce a highly compositional language and how accurately it manages communication tasks. By adjusting and testing various hyperparameters, we establish a benchmark for the model's performance before moving on to more complex experimental variations.

We considered the topological similarity a success if it converged above 0.8 with a clear upward trend. This indicated the emergence of a high-ρ language. For the baseline, some hyperparameters were not further tuned, such as speaker learning rounds, listener learning rounds, and interaction rounds. However, other critical hyperparameters, such as the knowledge pair size, were tuned to optimize performance, as discussed in subsection 4.2.4.

Vocabulary length and message length are the critical parameters we focus on; a more thorough investigation of these parameters is conducted in the subsequent sections. Here, we assume that the message length and vocabulary size for the baseline experiment were aligned with the initial language distribution. According to 4.2.4, the message length was set to 2, reflecting the composition of phrases consisting of a color and an object, and the vocabulary size was defined as n, corresponding to the items within the distinct sets.

The hyperparameters and their corresponding values in the baseline experiment are summarized in Table 4.1.

Table 4.1: Value of the Hyperparameter Used in the Baseline Experiment

Parameter	**Value**	**Description**
Learning Rate	1×10^{-3}	Step size during gradient descent
Selection Candidates	5	Number of candidates for the listener
Vocabulary Length	8	-
Message Length	2	-
Speaker pre-training Rounds	1200	Number of epochs of speaker's learning
Listener pre-training Rounds	200	Number of epochs of listener's learning
Interaction Rounds	3000	Number of epochs of interaction phase
Generations	80	Number of generations
Knowledge Pair	100	Number of object-message pairs sampled for speaker pre-training
Batch Size	64	-

4.2.3 Analysis of Result

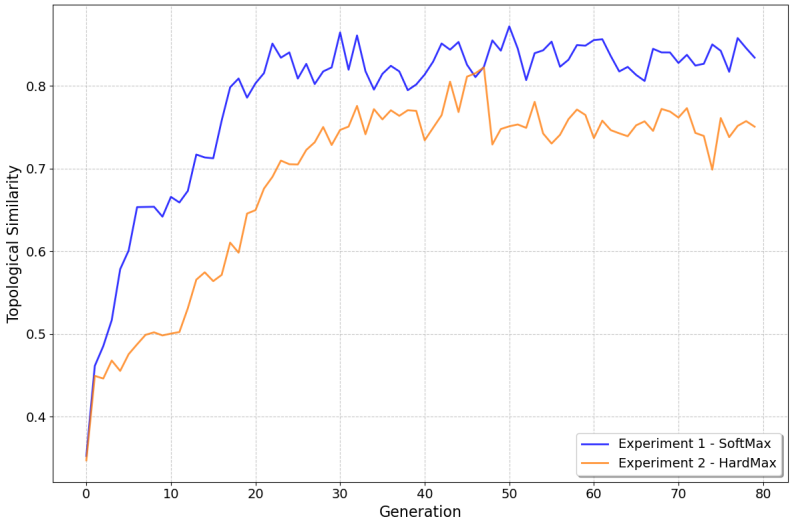


Figure 4.1: Topological Similarity of textual data

We set up the comparison experiments with different methods, one with SoftMax and the other one with HardMax. In the scencario of SoftMax, each class is assigned a probability from the classes distribution, from 0 to 1, summing up to 1. The values of probability represent different degrees of belief in each outcome candidate, and are therefore called "soft" for their non-binary property. However, the method HardMax makes a "hard" decision by transforming each output from SoftMax into a deterministic binary value - 1 assigned to the highest probability while the others 0. From Figure 4.1, we could tell that both setup show an overall upward trend of the topological similarity. However, SoftMax ensures a higher TS with faster convergence speed, which increases rapidly to above 0.8 within 20 generations and shows a relatively more stable performance afterwards, comparing with HardMax. The HardMax deterministic decision-making strategy lacks the flexibility for gradual adjustment, resulting in reduced adaptability to incorporate additional information and leading to a lower upper limit on optimal performance ultimately. Thus, we choose SoftMax for the later experiments.

We also print out the final language after 40 epochs of training, at which point the model has reached a stable state, as convinced by consistent rewards and accuracy. Provided that the emergent language has fully developed with steady structure, we could analyze the patterns and characteristics of the final language.

Below is one of the typical examples of the final language generated from our 8*8 dataset input in the first place (message length = 2, vocabulary size = 58, with 'Reinforce'). We observe that, for different objects of the same color, the coding is consistent across the columns. However, the pattern breaks down when considering the object of the same type but different colors. This phenomenon is universal for all of the experiments conducted, that the consistency in the color coding is always

higher than the object. Despite of some clusters shown under the same object, indicated a degree of regularity, the holistic uniformity for the same object still lacks.

	Cyan	Black	Orange	Pink	Red	Blue	Grey	White
Key	E 3	B 3	1 3	4 3	e 3	z w	U 3	V 3
Chair	E Z	B Z	1 Z	4 Z	e Z	z Y	U Z	V Z
Pen	E v	B v	1 v	4 v	e v	z v	U 5	V v
Ball	E a	B l	1 l	4 l	e l	z a	U l	V a
Watch	E B	B B	1 B	4 B	e e	z B	U B	V B
Laptop	E I	B I	1 q	4 I	e I	z I	U t	V p
Book	E O	B 5	1 O	4 O	e O	z 5	U 5	V N
Car	E H	B H	1 H	4 e	e e	z r	U H	V X

Table 4.2: A Typical Example of the Final Emergent Language

	Key	Chair	Pen	Ball	Watch	Laptop	Book	Car
Key	1.000	0.957	0.957	0.957	0.899	0.700	0.767	0.943
Chair	0.957	1.000	0.957	0.950	0.923	0.757	0.837	0.971
Pen	0.957	0.957	1.000	0.907	0.842	0.693	0.837	0.993
Ball	0.957	0.950	0.907	1.000	0.948	0.771	0.749	0.921
Watch	0.899	0.923	0.842	0.948	1.000	0.744	0.716	0.858
Laptop	0.700	0.757	0.693	0.771	0.744	1.000	0.590	0.729
Book	0.767	0.837	0.837	0.749	0.716	0.590	1.000	0.855
Car	0.943	0.971	0.993	0.921	0.858	0.729	0.855	1.000

Table 4.3: Cosine Similarity Matrix for the Second Symbol

4.2.4 Analysis of Knowledge Pair Size

This experiment examines the effect of varying knowledge pair sizes on speaker accuracy and topological similarity within the symbolic dataset. The symbolic data is constructed as described in Figure 4.2.4, where each unique phrase corresponds to a specific language description.

The size of the knowledge pairs, representing the set of object-message associations transmitted between generations, is a crucial hyperparameter in this setup. To ensure reliable experimental results for the symbolic data, we independently test different knowledge pair sizes, ranging from 20 to 160 in intervals of 20. The size of the knowledge pairs impacts the amount of information passed from one generation to the next, which, in turn, influences both speaker accuracy and the structural consistency of the emergent language, as measured by topological similarity.

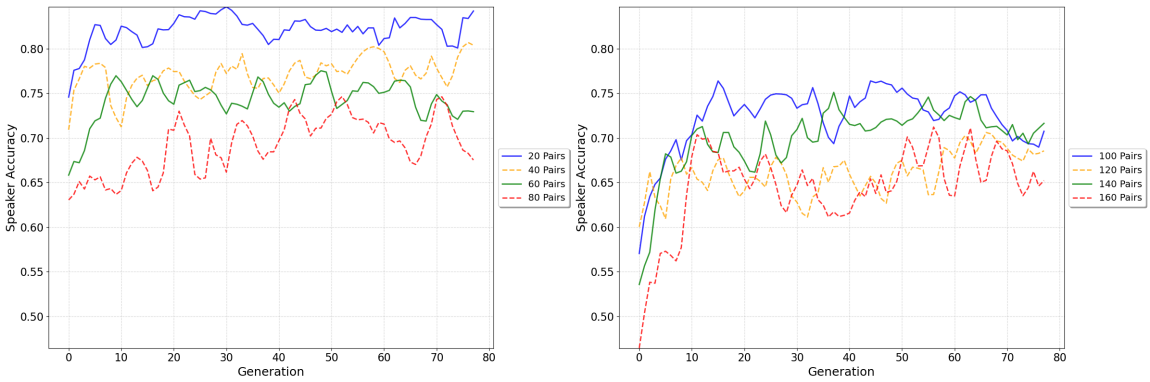


Figure 4.2: Speaker Accuracy on Symbolic Dataset with Smoothing Applied

In Figure 4.2, a smoothing function was applied to the data to better observe the trends. With multiple lines displayed and speaker accuracy often overlapping, smoothing reduces noise and provides a clearer view of the underlying patterns. This approach is commonly used to improve data interpretation by reducing random fluctuations (Smith and Doe, 2015). The function calculates a moving average within a specified window, where the average of neighboring points is taken for each data point. In this study, we selected a window size of 3 to strike a balance, ensuring that the smoothed data captures the main patterns without oversimplification.

The application of this smoothing function highlights that smaller knowledge pair sizes result in better speaker accuracy. This outcome is expected, as described in subsection 3.4.3, where speaker accuracy measures the rate of successful reproduction of the previous generation's message given the same input. A larger knowledge pair size introduces more tasks, making the task more challenging. Between 20 and 80 knowledge pairs, accuracy drops by approximately 0.05. From 100 to 160 pairs, each knowledge size shows a clear upward trend in the first 10 generations, followed by a similar decline, around 0.025. In contrast, for smaller knowledge sizes (20-80), speaker accuracy remains stable at a high value from the first generation onward, but this stability may suggest that the speaker is only superficially reproducing the messages rather than developing a deeper understanding of the transmitted knowledge across generation.

This observation suggests that iterated learning benefits from an appropriate increase in knowledge pair size within certain limits. To further explore this, we next examine topological similarity in Figure 4.3, by comparing topological similarity and speaker accuracy together, we observe that smaller knowledge pair sizes, such as 20, perform poorly, exhibiting significant fluctuations and low topological similarity. As the size of the knowledge pairs increases, a clearer upward trend in topological similarity emerges. This supports the earlier assumption that an appropriately higher knowledge pair size leads to improved performance in both topological similarity and speaker accuracy.

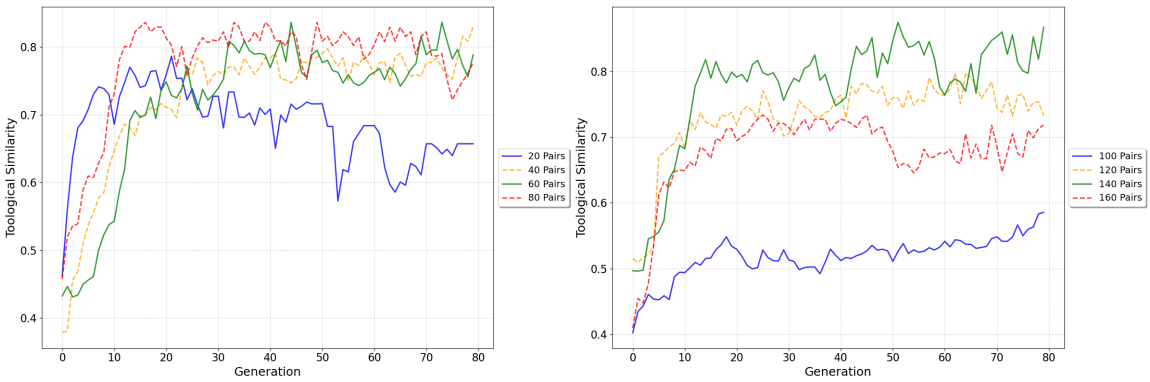
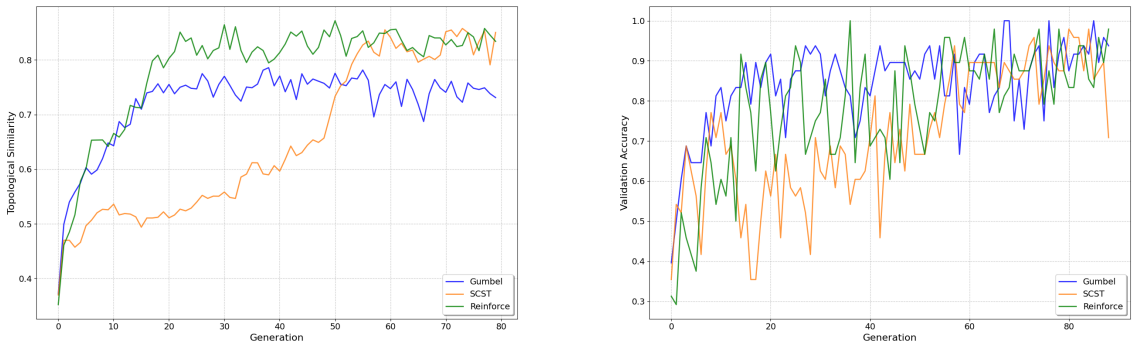


Figure 4.3: Topological Similarity on Symbolic Dataset

4.3 Comparison of different structures

- Reinforce vs Self-Critical Sequence Training vs Gumbel-Softmax

  Reinforce, Self-Critical Sequence Training (SCST), and Gumbel-Softmax (GS) are tested under the same arguments. The method Reinforce performs best with less effort for tuning the baseline (comparing with SCST) and higher accuracy (compared with GS). Figure 4.4a and 4.4b show their performance under the same argument setting.



(a) Topological Similarity (b) Accuracy

Figure 4.4: Comparison of Gumbel-Softmax vs SCST vs Reinforce

It is clear that with respect to TS, Reinforce achieved the highest topological similarity, maintaining above the level of 0.79 after 20 generations, which surpasses the optimum Gumbel achieved and the convergence speed of SCST. In terms of accuracy, however, these three methods differ little in their peak performance after 60 generations. Reinforce outperform in the long-term stability.

- Long Short-Term Memory vs Gated Recurrent Unit

Two commonly used variants of recurrent neural network (RNN) - Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) have been shown to perform well with long sequence applications (Dey and Salem, 2017). Despite GRU's advantage in computational efficiency with reduced gate networks, it performs poorly in our model with lower accuracy and significant oscillations. As illustrated 4.7, LSTM stably converged to the value above 0.9 within 20 generations of training while GRU struggled around 0.5 at the same time. The best performance of GRU is achieved after 75 generations, but it lacks steady convergence and continues to exhibit instability.

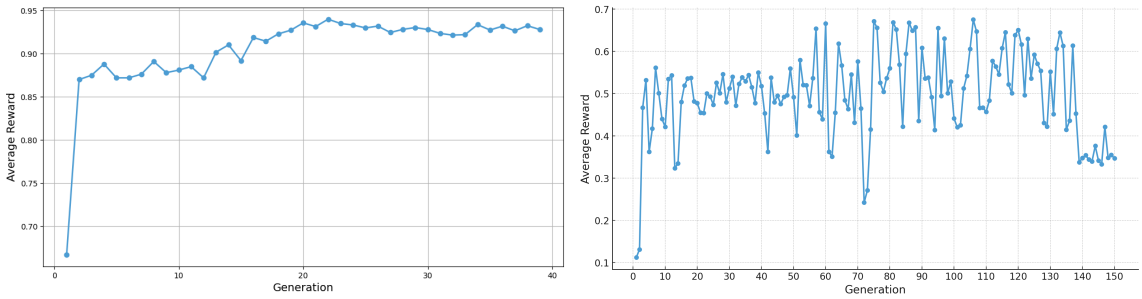


Figure 4.5: Average reward of LSTM(left) and GRU(right) throughout training

Given the input of symbolic stimuli, and the Reinforce method adopted, LSTM outperformed GRU overall, promoting us to use it for further experiments.

4.4 Effects of the message features

We investigate the effects of features of the both the vocabulary and the message.

- Vocabulary Length

  Our model's vocabulary consists of 84 characters, including 26 lowercase letters, 26 uppercase letters, 10 digits, and 16 special characters. (The vocabulary space is appended in Appendix A.) For an input of size of 8×8, it is sufficient to generate an emergent language with a subset of it. Thus, we examine the influence of the size of vocabulary onto the agents' communication, ranging from 8 to 78, in increments of 10, as shown in figure 4.6. We conducted 5 rounds of testing for each setting with 40 total epochs of running.

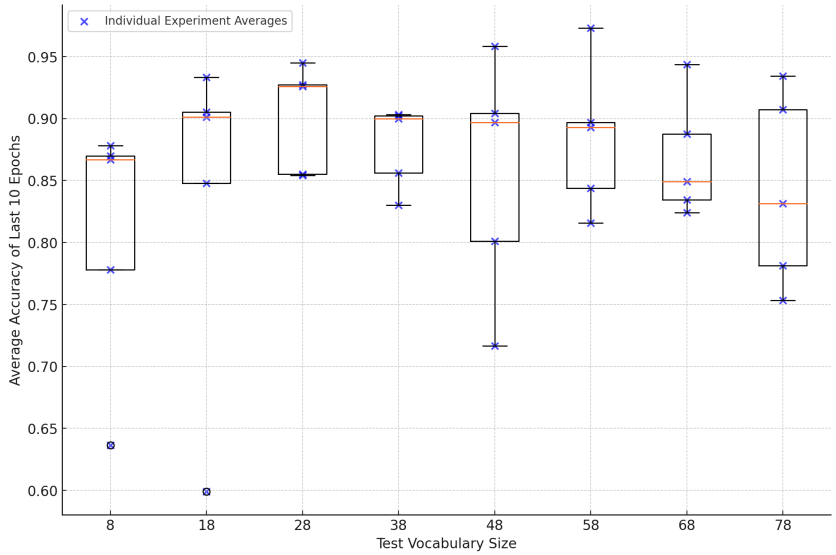


Figure 4.6: Last 10 Epochs' Average Accuracy Across Different Vocabulary Sizes

Although no linear correlation of the vocabulary size and the model's performance is observed, the peak of the accuracy climbs up till the experiment of $voc\_size = 58$. Experiment of $voc\_size = 28$ reaches the highest median accuracy value. These two vocabulary sizes achieve the optimal balance of sufficient symbols to utilize and not overwhelm the agents. Small vocabulary space such as that of 8 or 18 be inadequate in expression, thus, leading to a more spread out result. However, increasing the voc_size does not necessarily guarantee a better performance. As illustrated in the figure, with a voc_size of 78, the added complexity may hinder the agents' ability to make effective choices.

- Message Length

  As for the message length, we test out how the message length would affect the model's communication. Starting from the original setup with a message length of 2, we gradually increase the value, testing up to a message length of 10. It is clear that from the perspectives of average rewards and topological similarity, the optimal is achieved at the value 2, showing that the agents are capable to communicate efficiently with short messages. In another word, they could capture essential information with minimal symbols. However, as we increase the message length, the average rewards drops to 0.8 at $msg\_length = 10$. The clear downward trend in both graphs illustrates the importance of message length's role in our model, which leads to our next experiment.

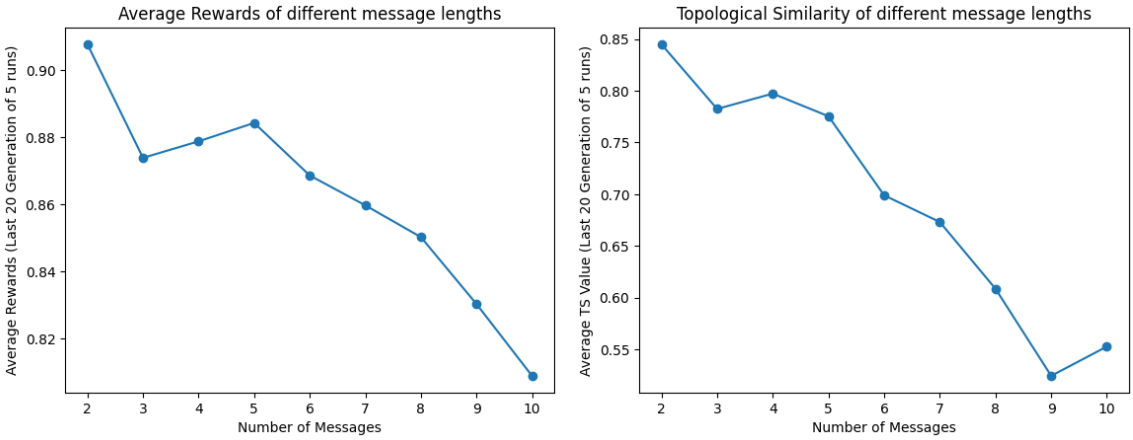


Figure 4.7: Last 10 Epochs' Average Accuracy(left) and Topological Similarity(right) Across Different Message Sizes

4.5 Penalty Experiment

4.5.1 The Game

To further investigate how the message length affects the performance of the agents' communication, we set up a NIL network with variable-length messages that transmit from the Speaker to the Listener agent. Upon the Speaker receiving the symbolic input, it produces a non-fixed length message to the Listener without any supervision. Past works have shown that, with the upper bound of the message length constrained, the communication system will develop to be anti-efficient, that the higher the frequency of use, the longer the corresponding length, always approaching to the setup threshold instead of towards brevity (Chaabouni et al., 2019). Here we aim to investigate if the same phenomenon will show up in our model with an NIL setup.

We testify this penalty experiment onto the same dataset we used for the previous textual experiment: a dataset consisting of the combinations of 8 distinct colors and 8 distinct objects, 64 possible entries produced in total.

4.5.2 Game Setup

The game is developed and enhanced using the EGG toolkit, which provides the primitives of variable-length sequences' communication (Chaabouni et al., 2019). The EGG framework is closely corresponding to Lewis's classic signaling game we introduced in 3.6 and allows for further development and exploration in other interesting research.

1. The Speaker takes an input i from one of the one-hot vectors. The input is randomly selected from the dataset we introduced in the prior part.

2. Speaker then pick a symbol sequence from the given vocabulary to form the message $M = \{m_1, m_2, ..., m_{s-1}, eos\}$ of length s, until 'eos', the end-of-sequence' token is generated by the Speaker or the maximum length of the message l_{max} is reached. In the latter case, 'eos' will be added to the message end. Therefore, every messages is ensured to suffix with 'eos' and length bounded by the maximum length l_{max}.

3. The Listener receives the final hidden state produced by RNN as the input associated with the 'eos' as the input and outputs a one-hot vector which denotes $\widehat{i}$. If the Listener's output $\widehat{i}$ is identical to the input Speaker receives i, it will be rewarded for the successful reconstruction.

4.5.3 Penalty Mechanism

An artificial regularizer punishing the message length is added onto the original loss:

$$L_1(i, L(msg), msg) = L_0(i, L(msg)) + \alpha \times \ |msg| \tag{4.1}$$

where $L_0(i, L(msg))$ denotes the original cross-entropy we used before and $|.|$ denotes the length. The hyperparameter α multiply the current length is the length cost to be added.

4.5.4 Results

We first conduct the penalty-free experiment, to investigate how the communication dynamics is as the baseline. Now we set the maximum message length at $l_{max} = 10$, as soon as a small penalty is introduced ($\alpha = 0.005$), an significant message length drop is observed. As the intensity of penalty increased from 0.0 to 0.20, the mean message length shows up the pattern as below 4.8. With the increase of the penalty coefficient α , the message length peaks at $\alpha = 0.02$ and gradually drops to around 3, which appears to be the optimum message length, or to say, the minimum message length required for effective communication for α set around 0.1. Experiments with penalty greater than that shows consistency in the training, where the beginning and final message length are mostly identical.

The topological similarity under each penalty setting is also shown in Figure 4.8. With the increase of length cost, the mean message length used throughout the communication is forced to decrease except for $\alpha = 0.02$, and accordingly, the topological similarity of the communication within the model increases with more information-compressed (shorter) messages.

An interesting finding is that, when the penalty is introduced with a relatively small value, i.e. $\alpha = 0.02$, the communication between agents are developing into an anti-efficient pattern. As in figure 4.9, the message incrementally lengthens throughout the training. This phenomenon also existed in Chaabouni et al. (2019)'s work, where longer messages are used for the agents' communication until length cost is introduced. Further researches can look into the mechanism behind it. We assume

this is caused by the lack of effort cost for transmission, unlike the communication pressure faced in nature language (Kirby et al., 2008), that the speaker tend to use as much information as possible to convey the knowledge, which will thus, lead to better performance.

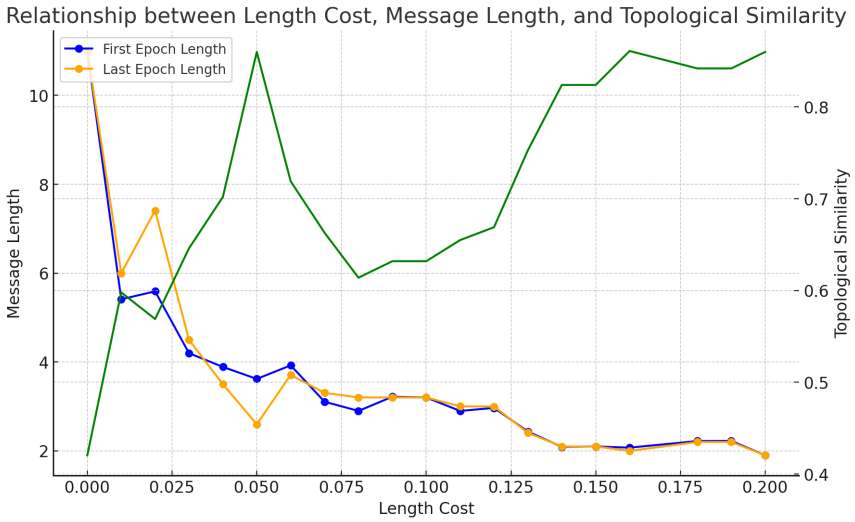


Figure 4.8: Message length vs the penalty coefficient α

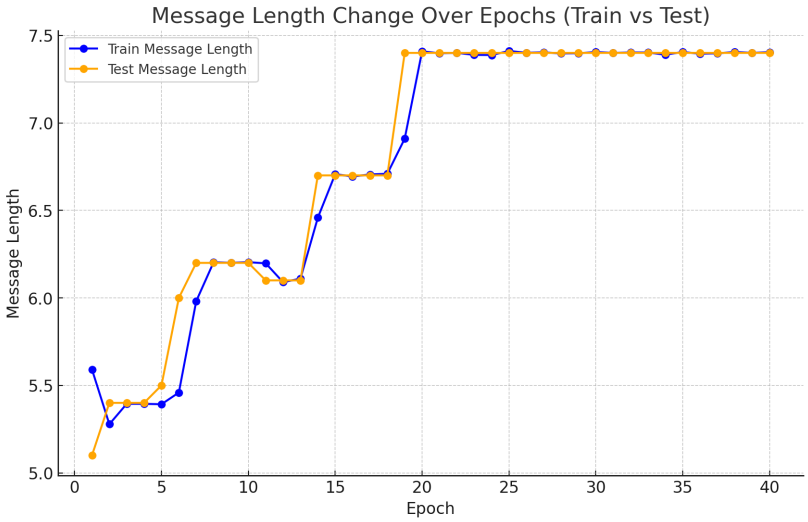


Figure 4.9: Message length under the penalty coefficient $\alpha = 0.02$

4.6 Discussion

The goal of this chapter was to explore how different elements, such as vocabulary and message length, influence the structure and effectiveness of emergent languages in our symbolic data systems. The previous experiments focused on two primary metrics: topological similarity and communication accuracy. These metrics allowed us to assess both the compositionality and the overall efficiency of the emergent communication system.

Baseline Experiment confirm the validity of the first hypothesis, demonstrating that Neural Iterated Learning, especially equipped with SoftMax is capable of developing a compositional language, with improving topological similarity throughout the training process. The use of SoftMax during the training phase resulted in higher and faster-converging topological similarity compared to HardMax, suggesting that soft probabilistic decision-making enhances the agents' ability to adapt and optimize the emergent communication system.

Structures Comparison Experiment further verified the inclusion that Reinforce and LSTM could enhance the overall performance of the model, both in the aspect of the compositionality of the emergent language and the performance stability.

Message Feature Experiment explored the impact of two important message features within the communication system. A relatively larger vocabulary size tends to enhance the optimal performance of the model, but if it becomes too large, agents struggle to make effective choices. However, an increase in message length reduces the model's accuracy until it surpasses a certain threshold. Once this value is exceeded, accuracy improves rapidly, but at the cost of significantly increased computational time.

Penalty Experiment shows that, as the penalty increases, the system stabilizes, and the message length converges to a minimal yet sufficient length for effective communication. This stability, alongside increasing topological similarity, indicates that the model finds an optimal balance between message length and communicative success when the penalty is appropriately set.

The topological similarity improves as the penalty for message length increases, except for the outstanding peak at the length cost of 0.05, where the final message length is relatively small. This means that a specific threshold of penalty is necessary to promote efficiency. The communication system develops into a more structured and efficient form as message length is penalized, with the agents learning to convey the same information in fewer symbols. This supports Hypothesis 4 that imposing a penalty encourages brevity and this finding aligns with Zipf's Law of Abbreviation.

In absence of the penalty, even under certain penalty setting i.e. $\alpha = 0.02$, the length of the messages that the agents used kept growing throughout the training process, which proves Hypothesis 5 that, an anti-efficient communication pattern could show up that longer messages are favored for more informative communication.

Chapter 5

Image Experiment

In this experiment, we extend our investigation to a more complex domain, transitioning from symbolic data to high-dimensional pixel-based input. This shift allows us to evaluate the model's ability to handle more intricate and visually rich data while maintaining the emergent of high ρ language observed in the simpler symbolic environment. The goal is to assess whether the agents can continue to perform effectively and whether topological similarity remains robust in this more challenging scenario.

5.1 Dataset

The dataset used in this study is derived from the MNIST digit images (Deng, 2012), which initially consist of 28×28 grayscale pixel arrays. To introduce an additional dimension of complexity, the images were colorized using a distinct palette, resulting in $28 \times 28 \times 3$ pixel arrays. This colorization process was applied uniformly across the selected digit classes to ensure consistency and avoid bias. The dataset was balanced by including an equal number of examples from each class. Subsequently, the images were converted into tensors with labels encoded as one-hot vectors.

Additionally, the original MNIST and colorized MNIST datasets were employed both separately and in combination to evaluate the effectiveness of Neural Iterated Learning in managing high-dimensional data, particularly when multiple attributes, such as digit shape and color, are involved.

5.2 Controls and Variables

The transition from symbolic data to image data, particularly hand-written MNIST, requires careful adjustment in hyperparameter settings due to the specific characteristics of image data. In symbolic datasets, each label is a distinct, unambiguous concept represented by a one-hot encoded vector. However, in the case of MNIST, which consists of hand-written digits, each label is associated with various images

that differ in stroke thickness and orientation, even though they represent the same digit. This inherent variability means the model requires adjustments beyond just evaluation methods like discussed in subsection 3.4.4.

This variability within the same label means the model must learn to recognize the underlying digit despite these differences. The hand-written nature of MNIST adds another layer of complexity, as individual hand-writing styles can vary significantly. For instance, two instances of the digit "3" may have different curvatures, line widths, or even slight distortions, yet they should still be classified under the same label. As a result, when transitioning from symbolic to image data like MNIST, hyperparameters such as learning rate, number of training rounds, message length, and vocabulary size need to be carefully tuned to ensure that the model can still converge to higher topological similarity.

For example, in the symbolic experiment, a batch size of 64 was sufficient because there were 64 distinct combinations of 8 colors and 8 objects, with each label uniquely corresponding to a specific combination. However, a larger batch size may be required in the MNIST dataset, where multiple images can correspond to the same label. This ensures that the agents have adequate exposure to the variability in the data, allowing them to learn more effectively across each epoch. The impact of batch size is explored in the subsequent experiments.

Moreover, it's important to consider that the type of dataset used—whether the original MNIST or the colored MNIST—significantly affects the number of unique labels. For instance, the original MNIST dataset consists of 10 labels corresponding to the 10 digits. However, when color is introduced, each digit can be associated with 10 different colors, resulting in 100 unique labels. This increase in label variety directly impacts the knowledge size required for training, as a larger label set necessitates a greater number of knowledge pairs for the speaker to learn from previous generations.

Considering these factors and the constraints related to running time, we tested a range of settings for key parameters like batch size, message length, and knowledge size. The hyperparameters and their corresponding values for the image experiments are summarized in Table 5.1.

5.3 Image Agent

We have designed an ImageAgent model for processing original MNIST images and colored MNIST images, as shown in Table 5.2.

The forward function of the model is described as follows:

1. Apply the first convolutional layer, followed by batch normalization and ReLU activation:

$$x \leftarrow \text{ReLU}(\text{BatchNorm}_1(\text{Conv}_1(x)))$$

Table 5.1: Value of the Hyperparameter Used in the Image Experiment

Parameter	**Value**	**Description**
Learning Rate	5×10^{-4}	Step size during gradient descent
Selection Candidates	5	Number of candidates for the listener
Vocabulary Length	60, 120	-
Message Length	2, 4	-
Speaker Learning Rounds	1500	Number of pre-training rounds for the speaker
Listener Learning Rounds	200	Number of pre-training rounds for the listener
Interaction Rounds	3000	Number of interaction rounds between agents
Generations	40, 80	Number of generations
Knowledge Pair	50, 100, 300	Number of object-message pairs sampled for speaker learning
Feature Size	200	The output feature size set in image agent
Batch Size	50	-

2. Apply the second convolutional layer, followed by batch normalization and ReLU activation:

$$x \leftarrow \text{ReLU}(\text{BatchNorm}_2(\text{Conv}_2(x)))$$

3. Apply the third convolutional layer, followed by batch normalization and ReLU activation:

$$x \leftarrow \text{ReLU}(\text{BatchNorm}_3(\text{Conv}_3(x)))$$

4. Apply adaptive average pooling to reduce the spatial dimensions:

$$x \leftarrow \text{AdaptivePool}(x)$$

5. Flatten the tensor to prepare it for the fully connected layer:

$$x \leftarrow \text{View}(x, -1, 128 \times 7 \times 7)$$

6. Apply the fully connected layer to obtain feature representation:

$$y_{\text{feature}} \leftarrow \text{FC}(x)$$

7. Apply the classifier to the feature representation to obtain the final output:

$$y \leftarrow \text{Classifier}(y_{\text{feature}})$$

Table 5.2: ImageAgent Model Architecture

Layer	Type	Configuration
1	Convolutional	in_channels=3, out_channels=32, kernel_size=3, padding=1
2	Batch Normalization	num_features=32
3	ReLU Activation	-
4	Convolutional	in_channels=32, out_channels=64, kernel_size=3, padding=1
5	Batch Normalization	num_features=64
6	ReLU Activation	-
7	Convolutional	in_channels=64, out_channels=128, kernel_size=3, padding=1
8	Batch Normalization	num_features=128
9	ReLU Activation	-
10	Adaptive Average Pooling	Output size=(7, 7)
11	Fully Connected	Input: $128 \times 7 \times 7$, Output: FEAT_SIZE
12	Classifier	Input: FEAT_SIZE, Output: CAT_SIZE

5.4 Game Setup

As discussed in section 3.3, the training involves the learning for the both speaker and listener, the interaction phase where agents play the referential game to push the emergent language towards high topological similarity, the transmission phase that collects knowledge for the next generation's speaker's training.

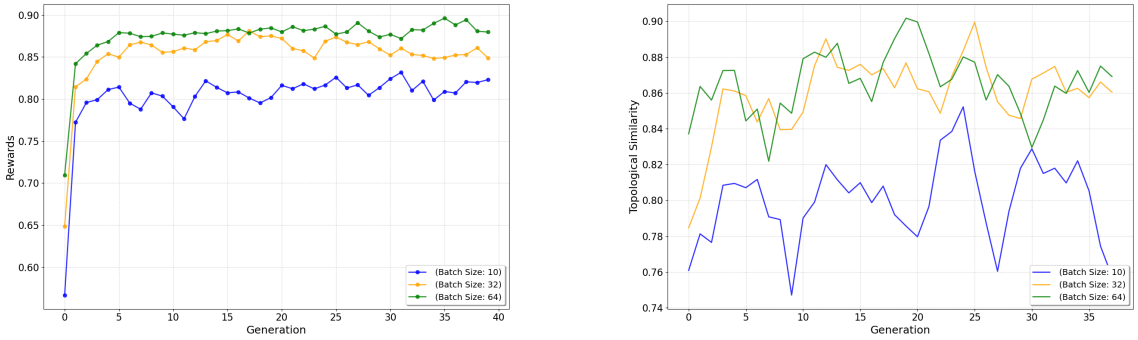
Experiment 1: Original MNIST With Raw Pixel Input

To build a general understanding of the functionality brought by the neural iterated learning to the high-dimensional pixel data, we first conduct experiments on the original MNIST dataset using the raw 28×28 pixel input, where each label containing 20 images.

In this experiment, following this initial setup, we investigated the effect of varying batch sizes on model performance. We began with a batch size of 10, equal to the number of labels, to establish a baseline. We then extended the batch sizes to 32 and 64, which exceeded the number of labels, allowing agents to recognize the variability within the same label. To enhance the visualization of the results, a smoothing function with a small window size of 3, as described in subsection 4.2.4, was applied to the topological similarity. Additionally, the reward was averaged over the interaction rounds of each generation to present an overall performance measure across generations, which is a standard approach in subsequent experiments.

Clearly, larger batch sizes result in improved rewards. Regarding topological similarity, there is a notable improvement when increasing the batch size from 1 to 32. However, both batch sizes of 32 and 64 exhibit high topological similarity during training, converging to ρ levels of 0.80, 0.86, and 0.87, respectively. Unlike the symbolic baseline experiment (Figure 4.1), where ρ grows steadily, significant fluctuations were observed in the growing of ρ, particularly when the batch size was smaller. This may be attributed to the difference in calculation methods; unlike the symbolic data where ρ is calculated multiple times and averaged within each generation, here it is computed once per generation and averaged across labels (subsection 3.4.4), due to the constraints of extended computational time. With smaller batch sizes, agents are exposed to less information for each digit, resulting in weaker generalization and more significant fluctuations in ρ.

Overall, larger batch sizes improve the model's generalization capacity across diverse intra-label variations. Although fluctuations in topological similarity persist with larger batch sizes, they are notably reduced compared to smaller batch sizes, reflecting an improvement in model stability. Nevertheless, the model's performance remains limited when processing pixel data, as the topological similarity still oscillates, albeit within a narrower range, highlighting the challenges posed by intra-class variability.



(a) Accuracy (b) Topological Similarity

Figure 5.1: Evaluation Matrix of Original MNIST With Raw Pixel Input

Experiment 2: Original MNIST With Extracted Feature Input

We extend our experiments to the original MNIST dataset using extracted feature input from image agent 5.3 to further investigate the impact of neural iterated learning. By transforming the high-dimensional pixel data into a lower-dimensional space, feature extraction simplifies the input while remaining essential information. Moreover, by leveraging higher-level abstractions such as edges and textures, the model can generalize more effectively across variations in the data, facilitating the emergence of highly compositional language.

The data structure in this experiment mirrors that of Experiment 1, with 10 labels and 20 inputs per label, using a feature size of 200. It is noteworthy that the image

agent achieved 97% accuracy on the classification task with test data after 200 epochs. The features were then extracted from the fully connected layer and used as input for the NIL model.

Compared to Experiment 1, this approach resulted in reduced rewards during the referential game, indicating a 10% decrease in successful communication between agents, which may be attributed to the increased complexity and richer information present in the raw pixel input. Despite these weaker rewards, topological similarity in this experiment demonstrated greater stability; a smoothing window size of 3 was applied as before. Among various batch sizes tested, batch size 64 yielded the best ρ, consistent with previous findings on batch size in experiment 1. Thus, we focus on this configuration for comparative analysis.

In this setting, ρ remained around 0.85 for the first 20 generations and exhibited an upward trend, reaching approximately 0.9 in the final 10 generations. This contrasts with Experiment 1, where ρ quickly attained 0.85 within the first generation but then fluctuated significantly and struggled to improve further. In Experiment 2, although ρ was slower to reach the 0.85 level, it displayed greater stability and a clear progression towards a high-ρ language.

These findings suggest that transforming data into lower-dimensional embeddings allows the model to capture generalized information more effectively, resulting in a more stable convergence towards a compositional language. This transformation helps to avoid distractions from lower-level features such as edges and textures, leading to a more consistent identification of high-ρ language. However, the reduced rewards are notable, as conventional expectations would be a positive correlation between high rewards and high topological similarity. To explore this anomaly further, we conducted additional comparisons focusing on message structure, detailed in section 5.5, to better understand this phenomenon's underlying dynamics.

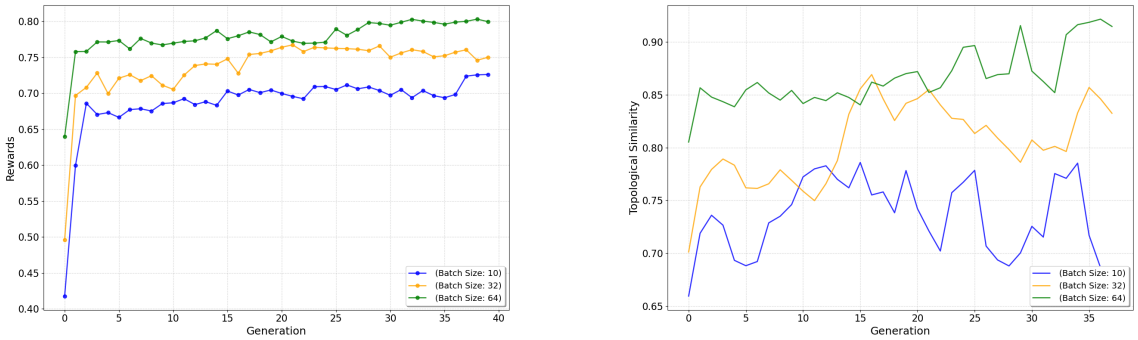
Moreover, compared to the **baseline symbolic data**, topological similarity in the image experiment converged significantly faster, reaching a stable higher level within the first 10 generations. In contrast, the symbolic experiment required approximately 30 generations to achieve a comparable level, even when considering that only 40 generations were used in the image experiment due to time constraints.

The rapid convergence observed in the image experiments, even when using raw pixel input, suggests that NIL agents are inherently more effective at developing compositional language with pixel data than symbolic data. This faster convergence may be attributed to the continuous nature of pixel representations, which provide a rich source of detailed information, enabling the model to form robust associations between input and output more quickly.

In the case of raw pixel inputs, the model can exploit spatial and intensity patterns inherent in the images, facilitating a more intuitive mapping between visual features and generated messages. This enables the agents to achieve a high degree of

compositionality in fewer iterations despite the increased complexity of the input data. In contrast, while conceptually simpler, the symbolic data needs this rich structure and requires more iterations to establish clear relationships, leading to slower convergence.

Overall, these findings underscore the robustness of NIL agents across varying data types and complexities. They can efficiently extract and utilize relevant patterns, achieving stable topological similarity even with complex input. This robustness reassures the audience about the model's reliability, making them feel confident in its performance.



(a) Accuracy (b) Topological Similarity

Figure 5.2: Evaluation Matrix of Original MNIST With Extracted Feature Input

Experiment 3: Colored MNIST With Raw Pixel Input

In this experiment, we explore the performance of neural iterated learning on the colored MNIST dataset using raw pixel input. By introducing color to the original grayscale MNIST images, the data now includes an additional dimension of complexity, with each image represented as a $28 \times 28 \times 3$ pixel array. This experiment aims to assess the model's ability to handle both shape and color attributes simultaneously, testing its capacity to process high-dimensional input data and capture the nuances of the added color information.

In this setup, we applied 10 distinct colors—Red, Green, Blue, Yellow, Cyan, Magenta, Orange, Royal Blue, Pink, and Turquoise—to the 10 digit classes, with each class containing 20 inputs, resulting in a total of 2,000 images.

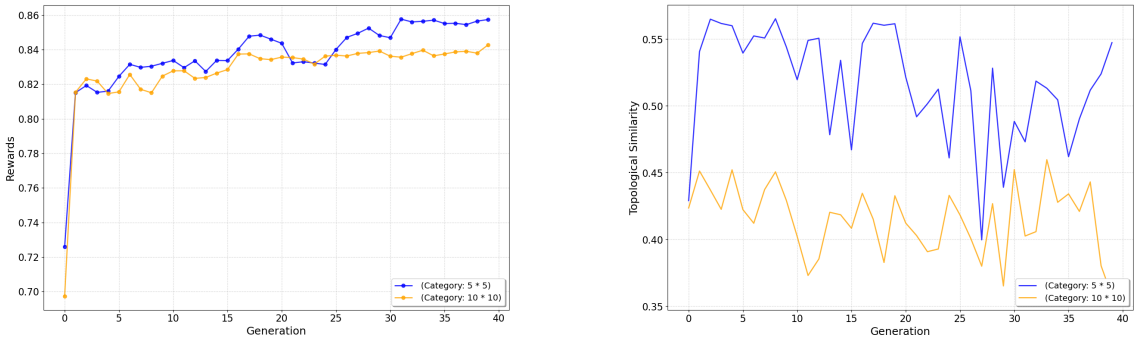
The results indicate that accuracy in this experiment was lower compared to Experiments 1, reaching only approximately 0.82. Topological similarity appeared significantly disrupted, fluctuating within the $0.40 \sim 0.55$ range. Although the pattern of topological similarity resembled that observed in Experiment 1, it was evident at a much lower level. Given the highly entangled dimensions in this experiment, it is likely that the speaker model was overwhelmed by the increased complexity of pixel data, coupled with the additional challenge of distinguishing colors. This

combination of factors appears to have hindered the emergence of a stable and high-ρ language.

To further investigate this issue, we utilized a smaller dataset containing only 5 colors and 5 digits, resulting in 500 images. As expected, this reduction led to improved topological similarity, achieving similar levels of accuracy. However, ρ remained low, with no indication of growth. We also tested varying batch sizes of 100, 200, and 300 of 10×10 data. Although there was a slight improvement in the topological similarity pattern with larger batch sizes, the overall level of ρ remained low, indicating that increasing batch size did not significantly enhance compositionality.

These findings suggest that NIL is significantly affected by the increased dimensionality and complexity, preventing it from converging to a high-compositional language despite achieving success in the referential game, as indicated by the reward scores. This could be attributed to the model's difficulty in managing the high variance and entanglement of visual features, which overwhelms its capacity to form structured communication.

Overall, while NIL can handle pixel inputs, the addition of color and increased data complexity hinders its ability to develop a robust and highly compositional language, demonstrating a limitation in its current form when applied to more intricate datasets.



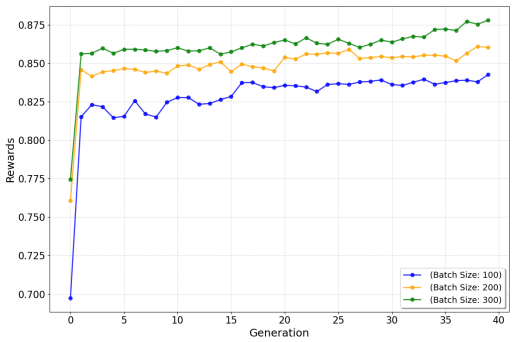
(a) Accuracy (b) Topological Similarity

Figure 5.3: Evaluation Matrix of Colored MNIST With Raw Pixel Input

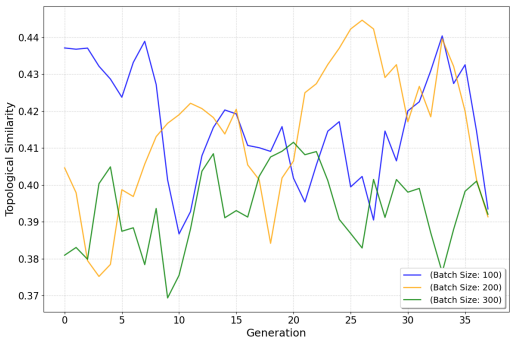
Experiment 4: Colored MNIST With Extracted Feature Input

Building on the previous experiments, this experiment evaluates the effectiveness of neural iterated learning on the colored MNIST dataset with extracted feature input.

For clarity, the image agent was tested using a validation set and achieved 93% accuracy on classification. The features, with a size of 200, were taken from the fully connected layer and used as data input. The dataset structure mirrors that of



(a) Accuracy

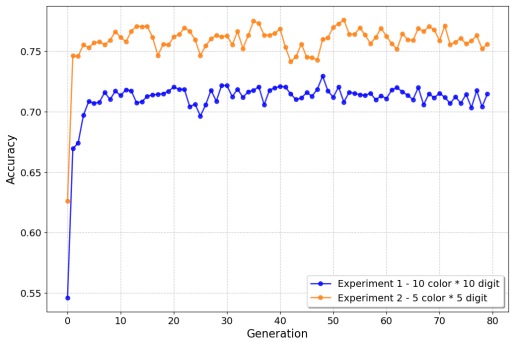


(b) Topological Similarity

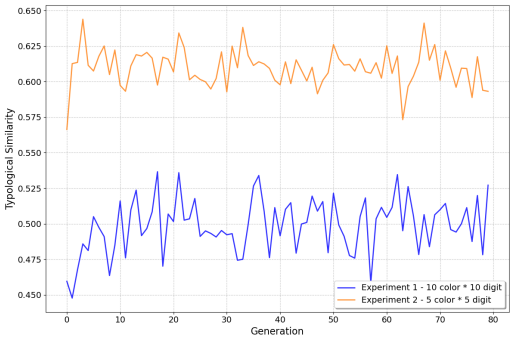
Figure 5.4: Evaluation Matrix of Colored MNIST With Raw Pixel Input of Batch Size

Experiment 3, with 10 colors and 10 digits. Additionally, a smaller subset of this dataset, containing only 5 colors and 5 digits, was also used for further testing.

We conducted a comparative evaluation similar to Experiment 3, maintaining identical hyperparameter settings across datasets with varying category diversity. Consistent with the findings from Figure 5.4a, we observed improved accuracy and topological similarity as the number of categories decreased. However, the topological similarity remained at a lower level, indicating a diminished generalization ability introduced by the increased complexity of the pixel data. Moreover, when applying the same hyperparameters, accuracy was still lower than in Experiment 3. This aligns with the comparison between Experiments 1 and 2, suggesting that the rich, detailed information provided by raw pixel input facilitates correct target identification, but also introduces complexity that can hinder communication efficiency.



(a) Accuracy



(b) Topological Similarity

Figure 5.5: Evaluation Matrix of Colored MNIST: Impact of Category Diversity on Extracted Feature Input

We then applied different batch sizes, which resulted in a clearer upward trend in topological similarity and improved rewards compared to Experiment 3. However, these gains were limited, with topological similarity stabilizing at a low range of $0.55 \sim 0.6$. To further explore performance optimization, we evaluated varying vocabulary sizes as shown in Figure 5.6. As vocabulary size increased from 60 to 200,

stability improved. Beyond a size of 120, topological similarity remained at a similar level but with greater stability, indicating that a larger vocabulary is required to maintain compositional language development in this more complex task compared to the original MNIST experiment.

We also analyzed the effect of message length and the number of attributes, as shown in Figure 5.7. The results show no significant impact on topological similarity with 2, 3, or 4 attributes. However, a notable drop occurred when the number of attributes increased to 5, with the best performance observed with 2 attributes (vocabulary size of 120). This suggests that the number of attributes is not a critical hyperparameter for the emergence of compositional language, and a configuration of 2 attributes is sufficient to achieve optimal performance.

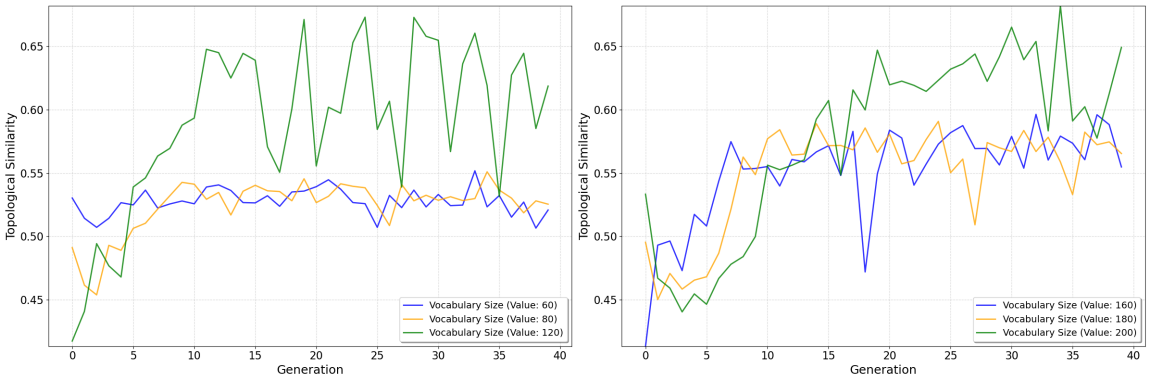
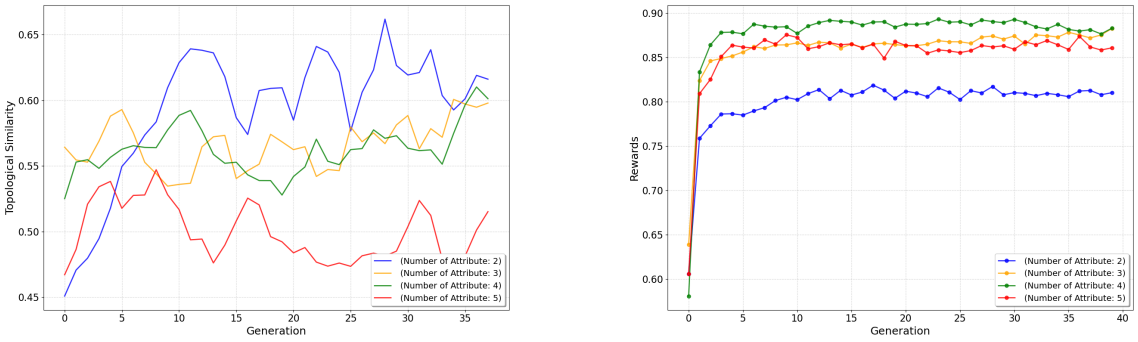


Figure 5.6: Topological Similarity of Colored MNIST: Impact of Vocabulary Size on Extracted Feature Input



(a) Accuracy (b) Topological Similarity

Figure 5.7: Evaluation Matrix of Colored MNIST: Impact of Message Length on Extracted Feature Input

In this experiment, we evaluated the effectiveness of neural iterated learning (NIL) on the colored MNIST dataset using extracted feature inputs. Although the image agent achieved 93% accuracy in a classification task, incorporating its features into NIL revealed a reduced generalization capability. As category diversity decreased, both accuracy and topological similarity improved, but overall topological similarity

remained low, highlighting the challenge posed by the complexity of pixel data. This complexity, while providing rich information, appears to hinder communication efficiency compared to more straightforward symbolic data. Adjusting batch sizes resulted in marginal gains in topological similarity and rewards but still fell short of reaching high compositional language levels. Vocabulary size was a critical factor, with stability improving as it increased from 60 to 200. However, beyond 120, further gains were minimal, indicating a threshold for adequate vocabulary size in this context. Similarly, varying the number of attributes revealed that using two attributes was optimal, while increasing to five significantly reduced topological similarity. These findings underscore that while NIL can handle high-dimensional image data, it requires careful optimization of vocabulary size and attribute configuration to develop stable and high-compositional languages in complex settings.

5.5 Message Shape

To further analyze the structure of the messages generated by the agents, we visualized the resulting message matrices using heatmaps, selecting a batch size of 64 due to its superior performance. These heatmaps depict the activation values of each vocabulary item across all digits, providing a detailed view of how different vocabulary elements are utilized to encode the features of each digit.

The heatmap is organized with vocabulary items as rows and digit representations as columns. Each cell corresponds to the aggregated activation value of a specific vocabulary item for a given digit. The color intensity of each cell, reflecting the magnitude of the activation value—with lighter hues indicating higher activations—helps identify key vocabulary items that play a crucial role in representing particular digits. This visualization is instrumental in understanding the underlying structure of the messages.

The heatmaps reveal that the messages are predominantly sparse, with sparsity levels ranging from 0.84 to 0.92 (Table 5.3), and 0.9 to 0.98 (Table 5.4) respectively in experiment 1 and 2. A detailed analysis reveals that, for both raw pixel input and extracted features, the speaker typically ar one or two vocabulary items per attribute for each unique image. Within the same label, different vocabulary items may be selected. Consequently, on average, each digit activates between 2-6 vocabulary items across the 20 images representing that label in the raw pixel input experiment and approximately 1-3 vocabulary items in the extracted feature input experiment.

Distinct activation patterns emerge across different digits, demonstrating that although the agent employs a sparse vocabulary, it effectively captures the unique features of each digit. The use of shared vocabulary elements across multiple digits suggests a compositional approach, where common message components are utilized to encode similar features. Conversely, the selective activation of specific vocabulary items for certain digits indicates the agent's capacity to encode distinguishing attributes uniquely. This balance between shared and unique components highlights the agent's ability to generalize common features while maintaining specificity in its

communication strategy.

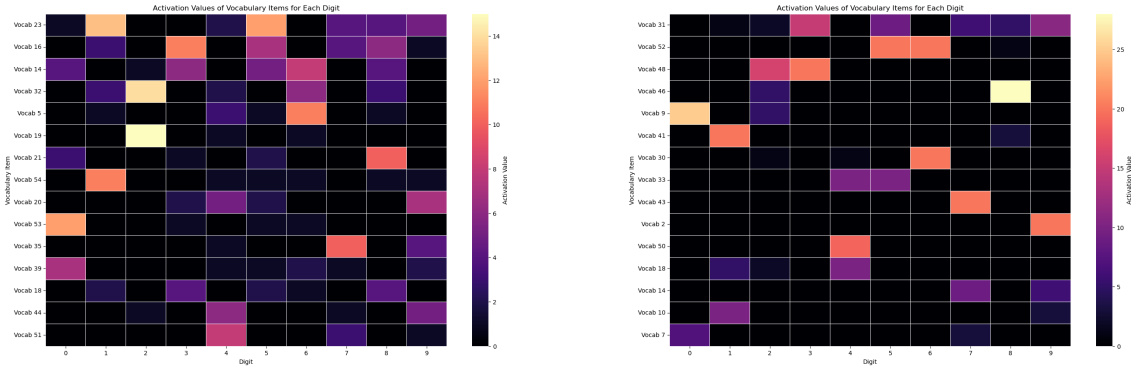
The higher sparsity observed in Experiment 2 aligns with our expectations that dimension reduction in pixel data results in more compressible language. This efficient use of vocabulary indicates that the agent has identified a compact set of features for distinguishing between pixel inputs. The elevated compositionality of the emergent language, as evidenced by higher ρ values, supports the hypothesis that agents, during training, tend to favor more compressible language structures (Kirby et al., 2015), a tendency further enhanced by the feature dimension reduction technique.

Digit	0	1	2	3	4	5	6	7	8	9
Sparsity	0.91	0.89	0.92	0.88	0.85	0.84	0.90	0.88	0.87	0.86

Table 5.3: Sparsity values in Experiment 1

Digit	0	1	2	3	4	5	6	7	8	9
Sparsity	0.95	0.96	0.90	0.96	0.97	0.97	0.98	0.96	0.95	0.97

Table 5.4: Sparsity values in Experiment 2



(a) Experiment 1

(b) Experiment 2

Figure 5.8: Message Heatmap

5.6 Discussion

In this study, we explored the application of neural iterated learning (NIL) to various configurations of the MNIST dataset, investigating raw pixel inputs and extracted feature representations across grayscale and colored images. The experiments aimed to evaluate the model's capability to develop compositional language and sustain high topological similarity under increasingly complex data scenarios.

Experiment 1 demonstrated that NIL can effectively handle high-dimensional pixel data from the original MNIST dataset, achieving significant accuracy and topological similarity, albeit with substantial fluctuations. The results indicated that larger batch sizes contributed to improved generalization and reduced variability in topological

similarity, although the model still faced challenges due to intra-class variations in the pixel data. Analysis of the message shape revealed that agents developed a sparse and efficient communication strategy, with a small subset of vocabulary items representing each digit.

Experiment 2 explored the impact of using extracted feature inputs from the same dataset, resulting in more stable topological similarity and enhanced accuracy. This highlighted the advantages of dimensionality reduction and feature abstraction, which help NIL focus on essential information while reducing noise from the high-dimensional input space.

Experiment 3 introduced the Colored MNIST dataset with raw pixel inputs, adding a layer of complexity through color attributes. This experiment revealed that the model struggled to maintain accuracy and topological similarity stability, likely due to the increased challenge of processing entangled color and shape information.

In **Experiment 4**, we used extracted features from the Colored MNIST dataset to address the challenges of raw pixel input. While this improved accuracy, especially with longer message lengths, topological similarity remained unstable, indicating that the added complexity of color and shape combinations continued to be a challenge for the model.

Overall, our findings suggest that while NIL can successfully generate compositional language in more straightforward settings, its performance is highly sensitive to the dimensionality and complexity of the input data. The introduction of color and the associated increase in data dimensionality complicated the learning process, often resulting in unstable topological similarity, supporting our hypothesis. Despite these challenges, the message shape analysis demonstrated that NIL is capable of developing efficient, structured communication strategies, with agents favoring more compact language forms to navigate complex data environments.

Chapter 6

Limitations and Further Work

Our study sheds light on understanding emergent language in the context of Neural Iterated Learning. However, it has certain limitations.

Firstly, the datasets used in this project are limited to small-scale synthetic textual data or commonly used off-the-shelf datasets. These do not fully capture the complexity and diversity of real-world data, where the variability is significantly greater. Additionally, the datasets employed are of optimal purity, meaning they are devoid of noise. This may result in the models lacking robustness and adaptability when applied to noisy, real-world data. Future research should explore a broader range of datasets, both in terms of genre and size, while also accounting for the increased computational cost and model complexity that such expansion would entail. Additionally, introducing noisy or imperfect data would help evaluate the model's performance in more realistic settings.

Secondly, due to time and computational budget constraints, the hyperparameters have not been entirely explored. For example, the symbolic model has at least 16 adjustable parameters, which means we have only scratched the surface in terms of tuning for optimal performance. A more exhaustive exploration of these hyperparameters, including learning rates, and message lengths could help ensure the final topological similarity consistently approaches 1 and potentially yield significantly better results. Future work should focus on systematic hyperparameter optimization to fully unlock the model's potential and ensure robustness across a wider range of tasks and datasets.

Thirdly, based on the results of the final language distribution, future research could investigate how different attributes are prioritized across the various layers of the model, which in turn affects the prioritization of different features. It would also be worthwhile to explore whether a more hierarchical structuring model could lead to greater consistency in feature representation and language formation. Examining how such structural adjustments influence the emergent language could provide deeper insights into improving both the compositionality and expressivity of the model's

communication system.

Another limitation of this study is the lack of a strictly parallel comparison between symbolic and pixel data. Although we hypothesize that the performance pattern of NIL remains consistent across these data types, differences in their structure necessitated modifications in code, such as changes to the evaluation methods and batch generation processes. These adjustments, driven by the inherent differences between symbolic and pixel data, mean that direct result comparisons are not strictly parallel. Consequently, our theoretical inferences are constrained, and further investigation is required to substantiate these observations.

Chapter 7

Conclusion

In this research, we investigate the NIL algorithm which is applied to both the symbolic dataset and the image dataset, such as MNIST, to reveal the nature of the emergent language with aspect to language compositionality and communication efficiency. Our experiments investigated the influence of key factors, including vocabulary size, message length, and penalization, on the emergent communication system. We show that the configuration of LSTM, Reinforce, and SoftMax works the best, which is used for the following experiments. Imposing a penalty on message length encourages brevity, resulting in more structured and efficient communication systems. This observation is consistent with Zipf's Law of Abbreviation and supports the hypothesis that penalization fosters concise and effective communication. However, an anti-efficiency pattern emerges under certain penalty settings, indicating that future research should investigate the underlying mechanisms behind this phenomenon.

The image experiments demonstrate that while NIL is capable of generating compositional language, its performance is heavily affected by the complexity and dimensionality of the input data, aligning with our hypothesis. We assumed that compressible representations would correlate with improved generalization, and indeed, in standard MNIST, NIL achieved strong topological similarity and accuracy, albeit sensitive to intra-class variations. This consistency across both symbolic and image datasets suggests robustness in NIL's performance, supporting our hypothesis of stable accuracy regardless of data type. However, as predicted, the model's ability to maintain compositional language diminished with increased data complexity, as seen in the Colored MNIST experiments, where fluctuating topological similarity occurred despite some accuracy gains with feature-extracted inputs. These findings emphasize that NIL's performance in high-dimensional contexts requires refined input representations and training setups to optimize generalization and stability.

These findings have important implications, demonstrating that NIL has strong potential for generating compositional languages in both structured symbolic data systems and more complex datasets. However, some limitations remain in our model. Future research should focus on improving NIL's robustness and generalization,

particularly in high-dimensional scenarios with increased complexity.

Bibliography

Jeffrey A. Barrett, Brian Skyrms, and Calvin Cochran. Hierarchical models for the evolution of compositional language, May 2018. URL `https://philsci-archive.pitt.edu/14725/`.

H Brighton and Simon Kirby. Understanding linguistic evolution by visualizing the emergence of topographic mappings. *Artificial Life*, 12(2):229–242, 2006. doi: 10.1162/106454606776073323.

Emil Carlsson, Devdatt Dubhashi, and Terry Regier. Cultural evolution via iterated learning and communication explains efficient color naming systems. In *Proceedings of the Annual Meeting of the Cognitive Science Society (CogSci)*, volume 45. Cognitive Science Society, 2023.

Jon W. Carr, Kenny Smith, Jennifer Culbertson, and Simon Kirby. Simplicity and informativeness in semantic category systems. *Cognition*, 202:104289, 2020. doi: 10.1016/j.cognition.2020.104289.

Rahma Chaabouni, Eugene Kharitonov, Emmanuel Dupoux, and Marco Baroni. Anti-efficient encoding in emergent communication, 2019.

In-Koo Cho and David M. Kreps. Signaling games and stable equilibria. *The Quarterly Journal of Economics*, 102(2):179–221, 1987.

Morten H. Christiansen and Nick Chater. Language as shaped by the brain. *Behavioral and Brain Sciences*, 31(5):489–508, 2008. doi: 10.1017/S0140525X08004998.

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. URL `https://doi.org/10.48550/arXiv.1904.09067`.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. doi: 10.1109/MSP.2012.2211477.

Rahul Dey and Fathi M. Salem. Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1597–1600, 2017. doi: 10.1109/MWSCAS.2017.8053243.

Gottlob Frege. über sinn und bedeutung. *Zeitschrift für Philosophie und philosophische Kritik*, 100:25–50, 1892.

Felix A. Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: continual prediction with lstm. *Neural Computation*, 12(10):2451–2471, 2000. doi: 10.1162/089976600300015015.

Thomas L. Griffiths and Michael L. Kalish. Language evolution by iterated learning with bayesian agents. *Cognitive Science*, 31(3):441–480, 2007. doi: 10.1080/15326900701326576.

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. *CoRR*, abs/1910.05291, 2019. URL `http://arxiv.org/abs/1910.05291`.

Richard W Hamming. Error detecting and error correcting codes. *The Bell system technical journal*, 29(2):147–160, 1950.

Charles F. Hockett. The origin of speech. *Scientific American*, 203:88–96, 1960.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: how do neural networks generalise?, 2020. URL `https://arxiv.org/abs/1908.08351`.

Theo M. V. Janssen and Thomas Ede Zimmermann. Montague Semantics. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2021 edition, 2021.

Jasmeen Kanwal, Kenny Smith, Jennifer Culbertson, and Simon Kirby. Zipf's law of abbreviation and the principle of least effort: Language users optimise a miniature lexicon for efficient communication. *Cognition*, 165:45–52, August 2017. doi: 10.1016/j.cognition.2017.05.001. Epub 2017 May 8.

Simon Kirby. *Learning, bottlenecks and the evolution of recursive syntax*, page 173–204. Cambridge University Press, 2002. doi: 10.1017/CBO9780511486524.006.

Simon Kirby and M Tamariz. Cumulative cultural evolution, population structure and the origin of combinatoriality in human language. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 377(1843):20200319, 2022. doi: 10.1098/rstb.2020.0319.

Simon Kirby, Hannah Cornish, and Kenny Smith. Cumulative cultural evolution in the laboratory: an experimental approach to the origins of structure in human language. *Proceedings of the National Academy of Sciences*, 105(31):10681–10686, 2008. doi: 10.1073/pnas.0707835105. Erratum in: Proc Natl Acad Sci U S A. 2017 Mar 21;114(12):E2544. Epub 2008 Jul 30.

Simon Kirby, Thomas Griffiths, and Kenny Smith. Iterated learning and the evolution of language. *Current Opinion in Neurobiology*, 28:108–114, 2014. doi: 10.1016/j.conb.2014.07.014.

Simon Kirby, Monica Tamariz, Hannah Cornish, and Kenny Smith. Compression and communication in the cultural evolution of linguistic structure. *Cognition*, 141:

87–102, 2015. ISSN 0010-0277. doi: 10.1016/j.cognition.2015.03.016. URL https://www.sciencedirect.com/science/article/pii/S0010027715000815.

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language, 2017. URL https://arxiv.org/abs/1612.07182.

Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. Emergence of linguistic communication from referential games with symbolic and pixel input, 2018. URL https://arxiv.org/abs/1804.03984.

Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8):707–710, 1966.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. 17, 04 2015.

David Kellogg Lewis. *Convention: A Philosophical Study*. Wiley-Blackwell, Cambridge, MA, USA, 1969.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019. URL https://arxiv.org/abs/1509.02971.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. doi: 10.1038/nature14236. URL https://doi.org/10.1038/nature14236.

Richard Montague. The proper treatment of quantification in ordinary english. In K.J.J. Hintikka, J.M.E. Moravcsik, and Patrick Suppes, editors, *Approaches to Natural Language*, volume 49 of *Synthese Library*. Springer, Dordrecht, 1973. doi: 10.1007/978-94-010-2506-5_10. URL https://doi.org/10.1007/978-94-010-2506-5_10.

Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. URL https://doi.org/10.1609/aaai.v32i1.11492.

R.M. Nefdt. A puzzle concerning compositionality in machines. *Minds & Machines*, 30:47–75, 2020. doi: 10.1007/s11023-020-09519-6. URL https://doi.org/10.1007/s11023-020-09519-6.

Barbara Hall Partee. *Compositionality in Formal Semantics: Selected Papers of Barbara H. Partee*. Blackwell Publishing, Malden, MA, 2004.

Billy Peralta, Tomás Sepúlveda, Orietta Nicolis, and Luis Caro. Space-time prediction of pm2.5 concentrations in santiago de chile using lstm networks. *Applied Sciences*, 12(22):11317, 2022. doi: 10.3390/app122211317. URL https://doi.org/10.3390/app122211317.

Y Ren, S Guo, L Qiu, B Wang, and DJ Sutherland. Language model evolution: An iterated learning perspective. *arXiv preprint arXiv:2404.04286*, 2024. URL https://arxiv.org/abs/2404.04286.

Yi Ren, Shangmin Guo, Matthieu Labeau, Shay B. Cohen, and Simon Kirby. Compositional languages emerge in a neural iterated learning model, 2020.

Yi Ren, Samuel Lavoie, Mikhail Galkin, Danica J. Sutherland, and Aaron Courville. Improving compositional generalization using iterated learning and simplicial embeddings, 2023.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7008–7024, 2017.

Hasim Sak, Andrew W Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. 2014.

John Smith and Jane Doe. Smoothing techniques for enhanced data visualization. *Journal of Data Analysis*, 12(3):123–134, 2015.

Michael Spence. Job market signaling. *The Quarterly Journal of Economics*, 87(3): 355–374, 1973.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018a. in preparation.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning, Second Edition: An Introduction*. The MIT Press, Cambridge, MA, 2018b. ISBN 9780262039246. URL `https://mitpress.mit.edu/9780262039246`. Includes 64 color illustrations and 51 black-and-white illustrations.

Ankit Vani, Max Schwarzer, Yuchen Lu, Eeshan Dhekane, and Aaron Courville. Iterated learning for emergent systematicity in vqa. In *International Conference on Learning Representations (ICLR)*, 2021a.

Ankit Vani, Max Schwarzer, Yuchen Lu, Eeshan Dhekane, and Aaron C. Courville. Iterated learning for emergent systematicity in VQA. *CoRR*, abs/2105.01119, 2021b. URL `https://arxiv.org/abs/2105.01119`.

Bill Waggener. *Pulse Code Modulation Techniques*. Springer, 1995. ISBN 978-0-442-01436-0. Retrieved 13 June 2020.

Kyle Wagner, James A Reggia, Juan Uriagereka, and Gerald S Wilkinson. Progress in the simulation of emergent communication and language. *Adaptive Behavior*, 11(1):37–69, 2003.

Wojciech Zaremba, Tomas Mikolov, Armand Joulin, and Rob Fergus. Learning simple algorithms from examples. In *International conference on machine learning*, pages 421–429. PMLR, 2016.

George Zipf. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, 1949.

Appendix A

Vocabulary Space

The full vocabulary used in our model is shown below, consisting of 84 characters, including 26 lowercase letters, 26 uppercase letters, 10 digits, and 16 special characters:

a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z,
A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z,
1, 2, 3, 4, 5, 6, 7, 8, 9, 0, ‚, !, @, #, $, %, ‚, &, *, (,), _, +, <, >, ?