COMPOSITIONAL LANGUAGES EMERGE IN A NEURAL ITERATED LEARNING MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

The principle of compositionality, which enables natural language to represent complex concepts via a structured combination of simpler ones, allows us to convey an open-ended set of messages using a limited vocabulary. If compositionality is indeed a natural property of language, we may expect it to appear in communication protocols that are created by neural agents via grounded language learning. Inspired by the iterated learning framework, which simulates the process of language evolution, we propose an effective neural iterated learning algorithm that, when applied to interacting neural agents, facilitates the emergence of a more structured type of language. Indeed, these languages provide specific advantages to neural agents during training, which translates as a larger posterior probability, which is then incrementally amplified via the iterated learning procedure. Our experiments confirm our analysis, and also demonstrate that the emerged languages largely improve the generalization of the neural agent communication.

1 INTRODUCTION

Natural language understanding (NLU), which is exemplified by challenging problems such as machine reading comprehension, question answering and machine translation, plays a crucial role in artificial intelligence systems. So far, most of the existing methods focus on building statistical associations between textual inputs and semantic representations, e.g. using first-order logic (Manning et al., 1999) or other types of representations such as abstract meaning representation (Banarescu et al., 2013). Recently, *grounded language learning* has gradually attracted attention in various domains, inspired by the hypothesis that early language learning was focused on problem-solving and tasks relevant to survival (Kirby & Hurford, 2002). While related to NLU, it focuses on the *pragmatics* (Clark, 1996) of learning natural language, as it implies learning language from scratch, grounded in experience. This research is often practiced through the development of neural agents which are made to communicate with each other to accomplish specific tasks (for example, playing a game). During this process, the agents build mappings between the concepts they wish to communicate about and the symbols used to represent them. These mappings are usually referred to as 'emergent language'.

So far, an array of recent work (Havrylov & Titov, 2017; Mordatch & Abbeel, 2018; Kottur et al., 2017; Foerster et al., 2016) has shown that in many game settings, the neural agents can use their emergent language to exchange useful coordinating information. While the best way to design games to favour language emergence is still open to debate, there is a consensus on the fact that we should gear these emergent languages towards sharing similarities with natural language. Among the properties of natural language, *compositionality* is considered to be critical, because it enables representation of complex concepts through the combinination of several simple ones. While work on incorporating compositionality into emergent languages is still in its early stage, several experiments have already demonstrated that by properly choosing the maximum message length and vocabulary size, the agents can be brought together to develop a compositional language that shares similarities with natural language (Li & Bowling, 2019; Lazaridou et al., 2018; Cogswell et al., 2019).

In a different body of language research literature, evolutionary linguists have already modelled the origins of compositionality for decades (Kirby & Hurford, 2002; Kirby et al., 2014; 2015). They proposed a cultural evolutionary account of the origins of compositionality and designed a framework called iterated learning to simulate the language evolution process, based on the idea that the

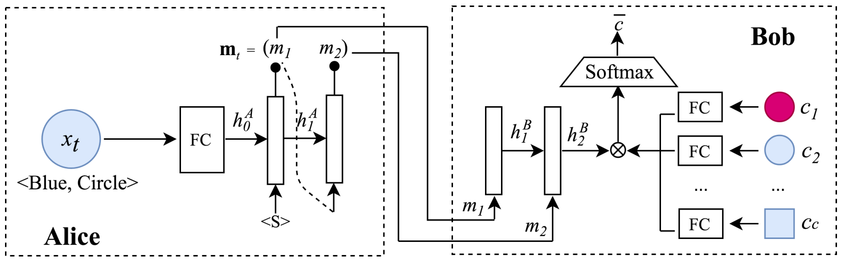


Figure 1: Referential communication game and architectures of the agents.

simulated language must be learned by new speakers at each generation, while also being used for communication. Their experiments show that highly compositional languages may indeed emerge through iterated learning. However, the models they introduced were mainly studied in the context of Bayesian agents and human participants in the experiment lab. When using Bayesian agents, these researchers designed a prior probability distribution that favors compositional languages[1]. In the case of experiments with human participants, it is argued that participants presumably favor those compositional languages because the human brain prefers structure. Although some early works attempted to combine iterated learning and neural network (Hurford et al., 1998; Batali, 1998; Kirby & Hurford, 2002), due to the specific design of the neural agents, directly applying this framework to grounded language learning is not straightforward.

In this project, we plan to build on the work of evolutionary linguistics and previous neural models by encouraging neural agents to invent highly compositional languages through iterated learning. To achieve this goal, we propose a three-phase neural iterated learning algorithm and a probabilistic explanation of it. The experimental results demonstrate that our algorithm can significantly enhance the topological similarity[2] between the emergent language and the original meaning space in a simple referential game (Lewis, 1969), without hindering the general game performance. Following our previous analysis, we confirm that as a result, neural agents have better generalization abilities, by highlighting the correlation between the topological similarity of the emergent language and the game performance in a zero-shot setting. Our contributions are as follows:

- We propose a three-phase iterated learning algorithm specifically adapted to neural agents.
- We propose a probabilistic approach to explain the mechanisms of our algorithm, allowing us to track the posterior distribution of the emergent languages used by the neural agents.
- We provide an analysis of the advantages offered by languages with *high topological similarity* for training both neural agents (speaker and listener), and explain how these advantages are gradually improved by the iterated procedure.

2 Background

In this section, we present the necessary background to our approach: the game to be played by the neural agents, the neural agents themselves, and the topological similarity, measure that we will use as a way to evaluate the compositionality of the emerging languages.

2.1 Referential Game

We analyze a typical and straightforward object selection game, in which a speaking agent (Alice, or speaker) and a listening agent (Bob, or listener) must cooperate to accomplish a task. In each round of the game, we show Alice a target object x_t selected from an object space $\mathcal{X}$ and let her send a discrete-sequence message $\mathbf{m}_t$ to Bob. We then show Bob c different objects (x_t must be one of them) and use $c_1, ..., c_c \in \mathcal{X}$ to represent these candidates. Bob must use the message received from

[1]Such languages can use different symbols to represent different attributes of meaning and combine these symbols in a systematic way to form a message such that the meaning of the whole message is formed from a simple combination of the meaning of its parts.

[2]This is a common measurement of language compositionality (Brighton & Kirby, 2006)

Alice to select the object that Alice refers among the c candidates. If Bob's selection $\bar{c}$ is correct, i.e. $\bar{c} = x_t$, then both Alice and Bob are rewarded. The objects are shuffled and candidates are randomly selected in each round to avoid the agents building mappings between the objects and their order of presentation.

In our game, each object in $\mathcal{X}$ has N_a attributes (color and shape are often used in the literature), and each attribute has N_v possible values. To represent objects, similarly to the settings chosen in (Kottur et al., 2017), we encode each attribute as a one-hot vector and concatenate the N_a one-hot vectors to represent one object. One simple example is the object space $\mathcal{X}$={blue box, blue circle, red box, red circle}, where $N_a = 2$ and $N_v = 2$.

The message delivered by Alice is a fixed-length discrete sequence $\mathbf{m} = (m_1, ..., m_{N_L})$, in which each m_i is selected from a fixed size meaningless vocabulary V. To ensure that the message can unambiguously describe all possible objects, we usually assume $N_L \geq N_a$ and $|V| \geq N_v$. For example, to work with the previously defined example $\mathcal{X}$, a vocabulary $\mathcal{V}$ containing only 2 symbols and a message length $N_L = 2$ would suffice.

2.2 Neural Agent Structures

Neural agents usually have separate modules for speaking and listening, which we name Alice and Bob. Their architectures, shown in Figure 1, are similar to those studied in (Havrylov & Titov, 2017) and (Lazaridou et al., 2018). Alice first applies a multi-layer perceptron (MLP) to encode x_t into an embedding, then feeds it to an encoding LSTM (Hochreiter & Schmidhuber, 1997). Its output will go through a softmax layer, which we use to generate the message $\mathbf{m}_t$. Bob uses a decoding LSTM to read the message and uses multiple MLPs to encode $c_1, ..., c_c$ into embeddings. Bob then takes the dot product between the hidden states of the decoding LSTM and the embeddings and applies a softmax layer to select the right object. As Alice and Bob are trained using reinforcement learning, we can use $p_A(\mathbf{m}_t|x_t; \theta_A)$ and $p_B(\bar{c}|\mathbf{m}_t, c_1, ..., c_c; \theta_B)$ to represent their respective policies, where θ_A and θ_B contain the parameters of each of the neural agents. When the agents are trained to play the game together, we use the REINFORCE algorithm (Williams, 1992) to maximize the expected reward under their policies, and add the entropy regularization term to encourage exploration during training, as explained in (Mnih et al., 2016). The gradients of the objective function $J(\theta_A, \theta_B)$ are:

$$\nabla_{\theta_A} J = \mathbb{E}\left[R(\bar{c}, x_t)\nabla \log p_A(\mathbf{m}_t|x_t)\right] + \lambda_A \nabla H[p_A(\mathbf{m}_t|x_t)] \tag{1}$$

$$\nabla_{\theta_B} J = \mathbb{E}\left[R(\bar{c}, x_t)\nabla \log p_B(\bar{c}|\mathbf{m}_t, c_1, ..., c_c)\right] + \lambda_B \nabla H[p_B(\bar{c}|\mathbf{m}_t, c_1, ..., c_c)], \tag{2}$$

where $R(\bar{c}, x_t) = \mathbb{1}(\bar{c}, x_t)$ is the reward function, H is the standard entropy function, and $\lambda_A, \lambda_B > 0$ are hyperparameters controlling regularization.

2.3 Measuring Compositionality

Compositionality is a crucial feature of natural languages, allowing us to use small building blocks (e.g., words, phrases) to generate more complex structures (e.g., sentences), with the meaning of the larger structure being determined by the meaning of its parts (Clark, 1996). However, there is no consensus on how to quantitatively assess the compositionality of a language. Besides a subjective human evaluation, *topological similarity* has been proposed as a possible quantitative measure (Brighton & Kirby, 2006).

To define topological similarity, we first need to define the languages studied in this work. A language $\mathcal{L}$ is a function mapping items from the object space $\mathcal{X}$ to the message space $\mathcal{M}$, i.e., $\mathcal{L}(\cdot) : \mathcal{X} \mapsto \mathcal{M}$. To compute topological similarity, we first need to measure the distances between pairs of objects: $\Delta_{\mathcal{X}}^{ij} = d_{\mathcal{X}}(x_i, x_j)$, where $d_{\mathcal{X}}(\cdot)$ is a distance in $\mathcal{X}$. Similarly, we compute the corresponding quantity for the associated messages $m_i = \mathcal{L}(x_i)$ in the message space $\mathcal{M}$ with $\Delta_{\mathcal{M}}^{ij} = d_{\mathcal{M}}(m_i, m_j)$, where $d_{\mathcal{M}}(\cdot)$ is a distance in $\mathcal{M}$. Then the topological similarity is defined as the correlation between these quantities across $\mathcal{X}$:

$$\rho(\mathcal{L}) \triangleq \frac{\sum_{x_i, x_j \in \mathcal{X}} \left(\Delta_{\mathcal{X}}^{ij} - \mathbb{E}[\Delta_{\mathcal{X}}]\right)\left(\Delta_{\mathcal{M}}^{ij} - \mathbb{E}[\Delta_{\mathcal{M}}]\right)}{\sqrt{\sum_{x_i, x_j \in \mathcal{X}} \left(\Delta_{\mathcal{X}}^{ij} - \mathbb{E}[\Delta_{\mathcal{X}}]\right)^2 \left(\Delta_{\mathcal{M}}^{ij} - \mathbb{E}[\Delta_{\mathcal{M}}]\right)^2}}, \tag{3}$$

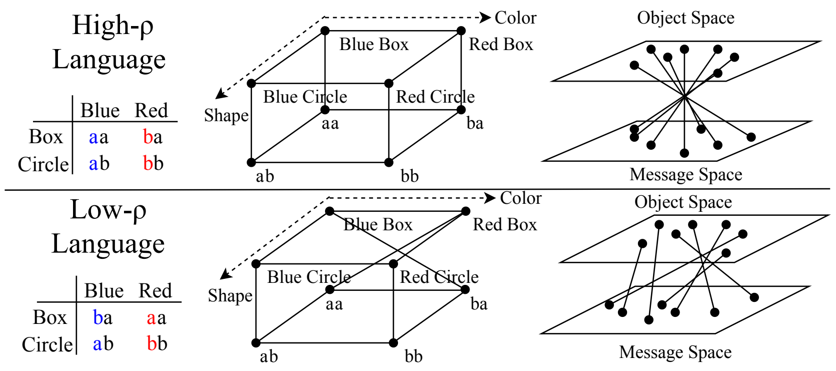


Figure 2: A simple representation of two languages corresponding to topological similarities of $\rho = 1$ (top) and $\rho = 0.5$ (bottom).

where $\mathbb{E}[\Delta_{\mathcal{X}}]$ and $\mathbb{E}[\Delta_{\mathcal{M}}]$ are the average distances among all possible pairs in each space. We should note that $\rho \in [-1, 1]$. Following the setup of (Lazaridou et al., 2018) and (Li & Bowling, 2019), we use the negative cosine similarity in the object space and Levenshtein distances (Levenshtein, 1966) in the message space.

To give a better intuition of what topological similarity is, we can apply it to a simple example: considering the previously introduced object space $\mathcal{X}$={blue box, blue circle, red box, red circle} and space of potential messages $\mathcal{M}$={aa,ab,ba,bb}, we show in Figure 2 an example of a high-ρ language (with $\rho = 1$) and a low-ρ language ($\rho = 0.5$). We can see that the mapping function of the high-ρ language is smoother and has less inflection points than the low-ρ language. That is because a high topological similarity implies that any pair of objects that are close to each other will have their mappings close in the message space.

3 Neural Iterated Learning Model

In iterated learning, the outputs of the agents training during one generation serve as input for training the agents in the next generation. This framework is proven to be effective when experimenting with both Bayesian agents and human participants. However, directly applying it to games played by neural agents is not trivial: for example, we cannot easily give high-ρ languages a large prior probability via a neural network. Besides, as Alice and Bob usually have different architectures, their way of obtaining information from the previous generation should be carefully designed. Hence in this section we propose an adapted iterated learning procedure where each generation is divided in three phases: the learning phase, the interacting phase, and the transmitting phase. We also provide a probabilistic analysis of our neural iterated learning algorithm, allowing us to tailor it to favoring high-ρ languages.

3.1 Neural Iterated Learning and Probabilistic Analysis

Our neural iterated learning algorithm for the referential game is detailed in Algorithm 1. The algorithm runs for I generations: at the beginning of each generation i, both the agents are re-initialized. They are then pre-trained using the data generated at the previous generation, which we denote D_{i-1}: this is the *learning phase*. As Alice and Bob have different structures, we pre-train them differently (see line 5-7 for Alice and line 8-12 for Bob): Alice is pre-trained via categorical cross-entropy and Bob is pre-trained with REINFORCE. We note I_a and I_b their respective number of pre-training iterations. Alice and Bob then play the game together for I_g rounds in the *interacting phase*, in which both agents are updated via REINFORCE. Finally, in the *transmitting phase*, we feed all objects to Alice and let it output the corresponding messages to be stored in D_i for the learning phase of the next generation.

To better understand how iterated learning can favor and let high-ρ languages dominate, we examine the posterior probability of all possible languages. Alice maps an object $x_t \in \mathcal{X}$ to a message $\mathbf{m}_t \in$

```
Randomly initialize $D_0$
for $i = 1, 2, ..., I$ do
  Re-initialize Alice and Bob, get Alice$_i$ and Bob$_i$
  // ======= Learning Phase =======
  for $i_a = 1, 2, ..., I_a$ do
    Randomly sample an example pair from $D_{i-1}$ and use it to update Alice$_i$ with
    cross-entropy training
  end for
  for $i_b = 1, 2, ..., I_b$ do
    Alice$_i$ generates message based on input objects
    Bob$_i$ receives message and selects the target
    Bob$_i$ updates its parameters if rewarded
  end for
  // ======= Interacting Phase =======
  for $i_g = 1, 2, ..., I_g$ do
    Alice$_i$ generates message based on input objects
    Bob$_i$ receives message and selects the target
    BOTH Alice$_i$ and Bob$_i$ update parameters if rewarded
  end for
  // ======= transmitting Phase =======
  for $i_s = 1, 2, ..., I_s$ do
    Generate object-message pairs by feeding all objects to Alice$_i$ and save them to data set
    $D_i$
  end for
end for
```

Algorithm 1: Our main neural iterated learning algorithm. D_i is the set of training pairs used to pre-train $Alice_i$ at generation i. I_a, I_b and I_g are the number of iterations used to pre-train Alice, Bob, and to play the game at each generation.

$\mathcal{M}$ via its softmax function. The message that is actually selected (and which mapping corresponds to the agents 'emergent language') corresponds to the argmax of this output. However, the other values allow us to compute the posterior distribution of a message given an input object $P(\mathbf{m}_t|D, x_t)$ by Alice, where D is the set of training samples it observed. Since languages are defined as mapping functions from $\mathcal{X}$ to $\mathcal{M}$, we can compute the posterior probability of any language using:

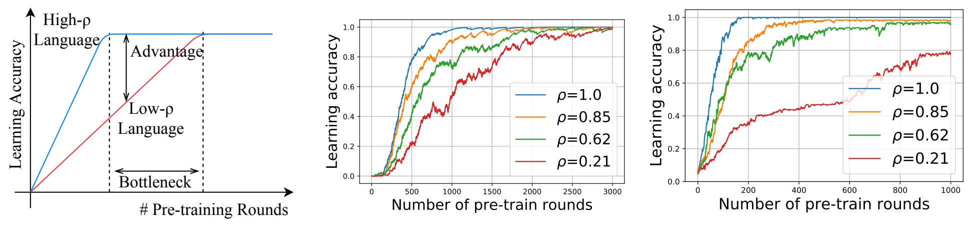
$$P(\mathcal{L}|D) = P(\mathbf{m}_1, ...|D, x_1, ...) = \prod_{\langle x_t, \mathbf{m}_t \rangle \in \mathcal{L}} P(\mathbf{m}_t|D, x_t), \tag{4}$$

assuming the mappings of different objects are independent. Denoting as $P_i(\mathcal{L})$ the posterior probability of a language $\mathcal{L}$ during the i-th generation, we can track the progress of any language during the iterated learning procedure.

While the distribution of languages is supposed to be close to uniform at the beginning of any generation i, pre-training the agents should bring the posterior probability of a language $\mathcal{L}$ to $P_i(\mathcal{L}|D_{i-1})$ (which is similar but not exactly the same with the distribution that generates D_{i-1}). New information is brought during the interacting phase, when the neural agents are only updated by successfully recognized objects-message pairs: we note this set of examples R_i. Then, the new posterior probability of $\mathcal{L}$ is $P_i(\mathcal{L}|D_{i-1}, R_i)$, and the set of examples for the next generation D_i is sampled from this new posterior distribution of languages.

From this analysis, we gain the intuition that posterior probability of languages represented by the object-message pairs in R_i would keep increasing over generations. We can already conjecture that ambiguous languages, that represent different objects with the same message, will necessarily see their probability decrease after a while. However, high-ρ languages only represent a small portion of all possible unambiguous languages (Brighton, 2002), even if this portion becomes relatively larger when N_L and $|V|$ are small[3]. While high-ρ language do not seem to be directly preferred during

[3]Hence, the agents have more chances to develop a high-ρ languages when N_L and $|V|$ are small, as illustrated by the experiments in (Lazaridou et al., 2018) and (Cogswell et al., 2019).



(a) Intuition of high-ρ advantage. (b) Experiments on Alice. (c) Experiments on Bob.

Figure 3: Illustration of the learning speed of Alice and performance improving speed of Bob when pre-training is done with various languages of different topological similarities. The game settings, network structures and hyperparameter settings are the same than for the experiments described in Section 4. We generate a perfect high-ρ language ($\rho = 1$) using the method proposed in (Kirby et al., 2015), and permute the messages to generate a low-ρ language with $\rho = 0.21$. The other two languages are intermediate languages generated during iterated learning.

the interacting phase, they can be favored by the neural agents during the learning phase, via careful setting of the hyperparameters of the pre-training procedure. We will make this explicit in the next two subsections.

3.2 Learning Speed Advantage for the Speaking Agent

In language evolution, highly compositional languages are favored because they are structurally simple and hence are easier to learn (Carr et al., 2017). We believe that a similar phenomenon applies to communication between neural agents:

Hypothesis 1: *High topological similarity improves the learning speed of the speaking neural agent.*

We speculate that high-ρ languages are easier to emulate for a neural agent than low-ρ languages. Concretely, that means that Alice, when pre-trained with object-message pairs describing a high-ρ language at a given generation, will be faster to successfully output the right message for each object. This principle is illustrated in Figure 3-(a). Intuitively, this is because the structured mapping described by a high-ρ language has a lower sample complexity, which makes resulting examples easier to learn for the speaker agent (Vapnik, 2013). To test our hypothesis, we observe the capacity of Alice to learn to reproduce object-message pairs corresponding to manually chosen languages during the learning phase. The experimental results confirm our hypothesis, as shown in Figure 3-(b).

This difference in learning speed will mechanically make the agent reproduce samples from high-ρ languages, which implies they will have higher posterior probabilities. As the training samples are randomly drawn from D_{i-1}, which we can interpret as a mixture of all possible languages weighted by their posterior probability at generation $(i-1)$, we translate our hypothesis as the following principle: the expected ρ of the set of possible languages in generation i should be higher than for generation $i-1$,

$$\mathbb{E}_{\mathcal{L}}[\rho(\mathcal{L})|D_i] \geq \mathbb{E}_{\mathcal{L}}[\rho(\mathcal{L})|D_{i-1}]. \tag{5}$$

As intuited in Figure 3-(a) and observed in Figure 3-(b), the gap between the two curves (which we call the 'advantage') varies with the number of training rounds. If the number of rounds I_a is too small, the agent will not have time to extract information from D_{i-1} well: the posterior distribution of languages will resemble its prior (uniform) version, which favors low-ρ languages. If I_a is too large, the agent will perfectly learn D_{i-1}, and equality will hold in equation 5: no improvement will be made. Computing an appropriate value for I_a is not trivial, since the learning speed depends on many other parameters, such as the learning rate, the optimizer, and the size of search space. We propose a simple approach, consisting in pre-training Alice with pairs describing a high-ρ and a low-ρ language, with all other hyper-parameters fixed. This gives us an approximate idea of the 'advantage' interval shown in Figure 3: we can then choose a value of I_a corresponding to this interval. Experiments specifically related to this issue are presented in Section 4.2

3.3 Learning Speed Advantage for the Listening Agent

In language evolution, highly compositional languages facilitate the description of new concepts, allowing using the attributes of the concepts that were previously learnt (Kirby et al., 2015). We believe a parallel can be drawn with how the structured mappings of high-ρ languages facilitate the task of learning to recognize these objects for the listening agent in our game:

Hypothesis 2: *High topological similarity allows the listening agent to successfully recognize more concepts, using less samples.*

Here, we speculate that high-ρ-languages are easier for a neural agent to recognize. That means that Bob, when pre-trained with message-object pairs corresponding to a high-ρ language, will be faster to successfully choose the right object: the recognition accuracy curves corresponding to a high-ρ and a low-ρ language for the listening agent should also exhibit the trend illustrated in Figure 3-(a).

Intuitively, the lower topological similarity is, the more difficult it will be to infer unseen object-message pairs from seen examples. The more complex mapping of a low-ρ language implies that more object-message pairs need to be provided to describe it (the worst case being all possible pairs but one). This translates as an inability for the listening agent to generalize the information it obtained from one object-message associated to a low-ρ language to other examples. Thus, the general performance of Bob on any example will improve much faster when trained with pairs corresponding to a high-ρ language than with a low-ρ language. To test this hypothesis, we randomly sample message-object pairs from languages associated to various values of ρ which we feed to four different instances of Bob, and observe their accuracy when recognizing objects. Again, the results match our hypothesis well: the high-ρ language indeed helps Bob to generalize faster than low-ρ languages, as shown in Figure 3-(c).

Similarly to I_a, the number of rounds I_b for which we pre-train the listening agent in the learning phase can be chosen by approximately finding the interval corresponding to this advantage, via separate pre-training using samples from a high-ρ language and a low-ρ language.

4 Experiments and Discussions

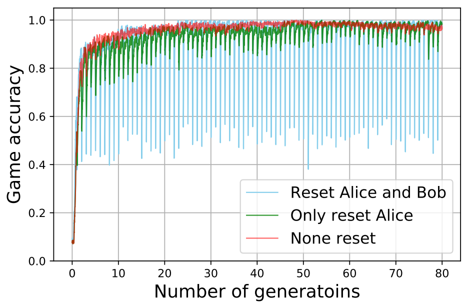
We now turn to an experimental study of the behavior of our proposed neural iterated learning algorithm. First, we examine the behavior and performance of the neural agents, as well as the posterior distribution of languages, at each generation. We conduct an ablation study, to examine the effect of pre-training Alice and Bob separately. We then investigate more thoroughly the advantages brought by high-ρ languages, and highlight the 'interval of advantage' in pre-training rounds described in Sections 3.2 and 3.3. Finally, we conduct a series of experiments on zero-shot playing of the object selection game, to highlight the positive effect of high-ρ languages on the neural agents generalization ability — which we believe shows the potential of iterated learning for NLU tasks. Details about our experimental setup and our choice of hyper-parameters can be found in the appendix.

4.1 Convergence Behavior

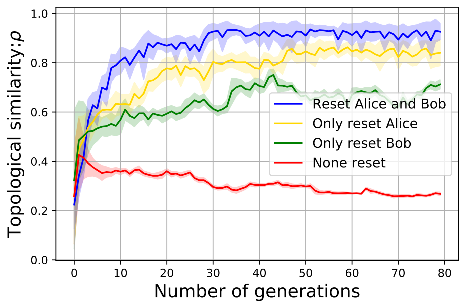
We are first interested in the evolution of the game performance and mean topological similarity during the iterated learning procedure, and if (and how) they converge. We set the number of rounds of the object selection game to be played in each generation at $I_g = 4000$ and set the number of generations to $I = 80$. We record the game performance (i.e., the rate of successful object selections) and mean ρ of the object-message pairs exchanged by the neural agents every 20 rounds. We run the simulation 10 times, with a different random number seed each time. Although the results are different, they all follow the same trend.

In this first series of experiments, we compare the following 4 different methods:

- Iterated learning, with resetting both Alice and Bob at the beginning of each generation.
- Iterated learning, only resetting Alice at the beginning of each generation;
- Iterated learning, only resetting Bob at the beginning of each generation;
- No iterated learning: neither Alice nor Bob are reset during training.



(a) Game performance



(b) Average ρ of emergent language

Figure 4: Game performance and average topological similarity for the possible resetting strategies of our proposed iterated learning procedure of 80 generations.

Let us first examine game performance: from Figure 4-(a), we can see that for the 3 displayed variants of the procedure, neural agents can play the game almost perfectly after a few generations[4]. The curve of the no-reset method will directly converge while the curves of the other two iterated learning procedures will show a loss of accuracy at the beginning of each generation. That is because one or both agents are reset, and are not able to completely re-learn from the data kept from the previous generation during the pre-training phase. However, at the end of each generation, all these algorithms can ensure a perfect game performance.

While the use of an iterated learning procedure has little effect on the game performance given a sufficient number of rounds, these procedures have a clear positive effect on topological similarity. In Figure 4-(b), we can see that the no-reset case has the lowest average ρ while the iterated learning cases all have higher means (and increasing). We will provide examples of the mixtures of languages associated to the no-reset and both-reset cases in Appendix D, showing that the high-ρ language exhibits a clear structure. We also examine in detail the posterior probability of languages according to their topological similarity, which is illustrated in Appendix D, confirming our current observations and providing a detailed account of the inner working of our procedure. We leave the discussion on the specific impact of each agent and why the reset-Alice and reset-Bob behave differently for Section 5.

4.2 High Topological Similarity and Interval of Advantage

In this section, we explore further the phenomenon described in Section 3.2 and 3.3. First, we take a look at the resulting topological similarity of the language sampled by Alice when pre-trained with languages corresponding to various initial ρ, depending on the number of pre-training rounds I_a. Results are displayed in Figure 5-(a). We can see that the same interval appears: the average topological similarity of the sampled language first increases, and then converges to a value close to the initial one.

Then, we examine the behavior of 3 different metrics for full runs of the iterated learning algorithm, for various values of I_a and I_b, while all other hyperparameters are fixed:

- $\mathbb{E}[r_{71:80}]$: The average reward of the last ten generations;
- $\mathbb{E}[\rho_{1:10}]$: The average value of ρ for the first ten generations;
- $\mathbb{E}[\rho_{71:80}]$: The average value of ρ for the last ten generations.

From the results presented in Table 1, we can see the importance of the number of pre-training rounds not being too large nor too small. The suitable I_a and I_b are shown in bold. From Figure 3, 5, and Table 1, we can see that the interval of advantage of I_a lies between 1000 to 2000 while it lies between 100 to 200 for I_b.

[4]We do not display the last case to avoid cluttering the figure.

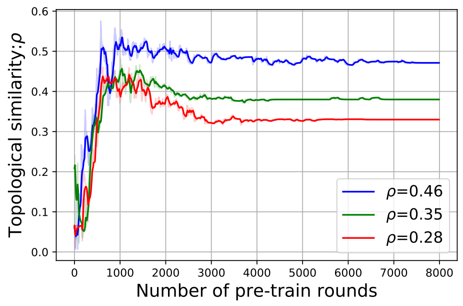


Figure 5: Transformation of ρ: we display the topological similarity of the language sampled by Alice depending of the number of pre-training rounds and the topological similarity of pre-training examples.

I_a	100	200	400	800	**1200**	**1500**	**2000**	3000	5000	8000
$\mathbb{E}[r_{71:80}]$	0.293	0.828	0.928	0.951	0.958	0.961	0.952	0.956	0.955	0.949
$\mathbb{E}[\rho_{1:10}]$	0.225	0.429	0.452	0.483	**0.556**	**0.575**	**0.566**	0.494	0.481	0.443
$\mathbb{E}[\rho_{71:80}]$	0.203	0.706	0.836	0.886	0.899	0.935	0.936	0.929	0.889	0.837
I_b	10	20	40	80	**120**	**160**	**200**	300	400	800
$\mathbb{E}[r_{71:80}]$	0.954	0.946	0.961	0.954	0.962	0.959	0.962	0.957	0.961	0.944
$\mathbb{E}[\rho_{1:10}]$	0.415	0.381	0.488	0.496	**0.591**	**0.535**	**0.557**	0.498	0.488	0.448
$\mathbb{E}[\rho_{71:80}]$	0.927	0.937	0.929	0.928	0.936	0.891	0.888	0.897	0.891	0.880

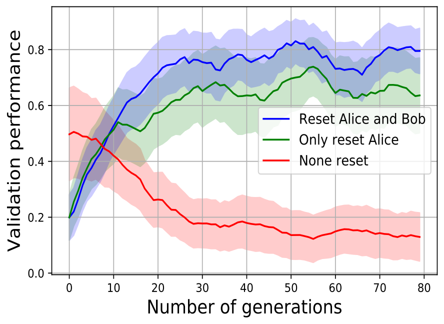
Table 1: Values of 3 metrics when varying I_a or I_b, highlighting an interval where the topological similarity grows high. Compositionality is maximised when the amount of data is neither too high nor too low.

4.3 Topological Similarity and Zero-Shot Performance

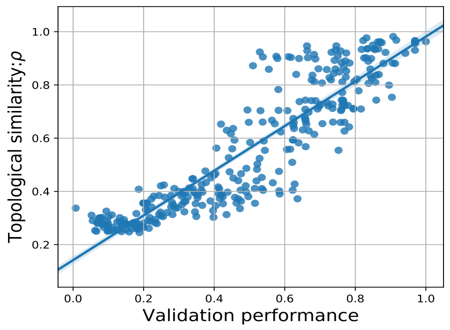
In this last series of experiments, we aim to explore the relationship between topological similarity and the generalisation ability of our neural agents, which can also indirectly reflect the expressivity of a language. We measure this ability by looking at their zero-shot game performance: we restrict the training examples to a limited numbers of objects (i.e., the training set), and look at how good are the agents at playing the game on the others (i.e., the validation set). Figure 6-(a) demonstrates the strength of the iterated learning procedure in a zero-shot setting. To illustrate the relationships between ρ and zero-shot performance, we randomly choose $I_a \in [60, 4000]$ and $I_b \in [5, 200]$ and conduct a series of experiments. As many experiments do not have optimal I_a and I_b, they will yield a worse performance on both zero-shot test and topological similarity. In Figure 6-(b), we record the results from different experimental settings and draw the zero-shot performance given the topological similarity of the emergent language. This shows the linear correlation between these two metrics, and a significance test confirms it: the correlation coefficient is 0.928, and the associated p-value is $3.8 * 10^{-104}$. Hence, under various experimental settings, the zero-shot performance and the topological similarity are strongly correlated. Table 2 illustrates that when the size of validation set increases, using iterated learning can always improve the zero-shot performance: in all the cases, both-reset algorithm always yields the best performance comparing with others. The fact that Alice-reset setting performs better than Bob-reset setting also matches our analysis well.

Valid set size	0		8		16		32	
	Train	Valid	Train	Valid	Train	Valid	Train	Valid
No-reset	0.985	-	0.986	0.136	0.990	0.132	0.995	0.102
Bob-reset	0.967	-	0.943	0.094	0.962	0.152	0.947	0.116
Alice-reset	0.981	-	0.976	0.598	0.979	0.280	0.947	0.210
Both-reset	0.988	-	0.986	0.847	0.984	0.755	0.973	0.558

Table 2: Zero-shot performance under different validation set size.



(a) Zero-shot performance, validation set size is 8.



(b) Results of zero-shot performance and ρ.

Figure 6: Zero-shot performance and topological similarity with validation size equals eight. Iterated learning leads to the evolution of languages which allow agents to perform well on unseen items.

5 Discussion: a parallel with Language Evolution

We can observe an interesting phenomenon in Figure 4-(b)[5]: the topological similarity of the emergent language always increases at first, whether we use iterated learning or not. This is akin to the effect apparent for ρ in Figure 5: continuing training will imply fine-tuning to examples that are not necessarily of good quality. However, through the generational resets and limited number of pre-training examples, iterated learning allow small generational improvements: this is because constraining the agent to learn with smaller amounts of data at each generation — through a 'bottleneck' (Kirby & Hurford, 2002) — forces the emergence of a more structured language. This limitation on the amounts of data available corresponds in our algorithm to limiting the number of pre-training rounds of the agents, to a number in what we denoted as the 'interval of advantage'. Extending this parallel with the evolution of natural language, we can relate the learning speed advantage provided by high-ρ languages to the speaking agent (from Section 3.2) to the *compressibility pressure* (Kirby et al., 2015), and the better ability to generalize provided by high-ρ languages to the listening agent (from Section 3.3) to the *expressivity pressure* (Kirby et al., 2015). This comparison allows us to address one important difference between our neural iterated learning algorithm and the original version: our speaking and listening agents are not identical. From Figure 4-(b) and Figure 6-(a), it is clear that Alice and Bob are affected differently by the generational resets, and thus do not offer the same contribution to the final performance[6]. From this parallel, we retain that iterated learning is also linked to the emergence of a certain form of compositionality when applied to neural agents. Besides, we believe that the correlation between topological similarity and zero-shot performance that we highlight in Section 4.3 is another argument in favor of a relationship between compositionality and generalization, which has recently been explored (Kottur et al., 2017; Choi et al., 2018; Andreas, 2019).

6 Conclusion

In this paper, we propose a neural iterated learning algorithm to encourage the dominance of high compositional language in a multi-agent communication game. We show that our procedure, consisting in resetting neural agents playing a referential game and pre-training them on data generated by their predecessors, can incrementally advantage emergent languages with high topological similarity. We demonstrate its interest by obtaining large performance improvements in a zero-shot game-playing setting, linking compositionality and ability to generalize to new examples. In the future, we plan to explore alternative pre-training strategies for the neural agents, and to extend our procedure to more complex neural-agents-based systems.

[5]This can be viewed in more details when looking at the probabilistic analysis presented in the appendix

[6]However, this parallel may not explain how differently they contribute to gains in topological similarity, since we must factor in the differences between their pre-training procedures, and especially the fact that Alice is pre-trained by minimizing cross-entropy.

REFERENCES

Jacob Andreas. Measuring compositionality in representation learning. *arXiv preprint arXiv:1902.07181*, 2019.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pp. 178–186, 2013.

John Batali. Computational simulations of the emergence of grammar. *Approach to the Evolution of Language*, pp. 405–426, 1998.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Henry Brighton. Compositional syntax from cultural transmission. *Artificial life*, 8(1):25–54, 2002.

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

Jon W Carr, Kenny Smith, Hannah Cornish, and Simon Kirby. The cultural evolution of structured languages in an open-ended, continuous world. *Cognitive science*, 41(4):892–923, 2017.

Edward Choi, Angeliki Lazaridou, and Nando de Freitas. Compositional obverter communication learning from raw visual input. In *International Conference on Learning Representations*, 2018.

Herbert H Clark. *Using language*. Cambridge university press, 1996.

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.

Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 2137–2145, 2016.

Serhii Havrylov and Ivan Titov. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. In *Advances in Neural Information Processing Systems*, 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

James Raymond Hurford, James R Hurford, Michael Studdert-Kennedy, and Chris Knight. *Approaches to the evolution of language: Social and cognitive bases*. Cambridge University Press, 1998.

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

Simon Kirby, Tom Griffiths, and Kenny Smith. Iterated learning and the evolution of language. *Current opinion in neurobiology*, 28:108–114, 2014.

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

Satwik Kottur, José Moura, Stefan Lee, and Dhruv Batra. Natural language does not emerge naturallyin multi-agent dialog. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2962–2967, 2017.

Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. Emergence of linguistic communication from referential games with symbolic and pixel input. In *International Conference on Learning Representations*, 2018.

Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pp. 707–710, 1966.

David Lewis. *Convention: A philosophical study*. John Wiley & Sons, 1969.

Fushan Li and Michael Bowling. Ease-of-teaching and language structure from emergent communication. *arXiv preprint arXiv:1906.02403*, 2019.

Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937, 2016.

Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

APPENDIX A: PARAMETER SETTINGS

Unless specifically stated, the experiments mentioned in this paper use the hyper-parameters given in Table 3. The code used to realise our experiments will be released upon publication.

Notation	Value	Description
N_a	2	Number of all attributes
N_v	8	Number of possible values for each attribute
N_L	2	Message length
$\|V\|$	8	Vocabulary size. (Here we have $V = \{\text{abcdefgh}\}$)
I	80, 100	Maximum number of generations
I_a	$\geq 100, \leq 8000$	Maximum pre-train **rounds** for Alice
I_b	$\geq 10, \leq 800$	Maximum pre-train **batches** for Bob
I_g	$\geq 100, \leq 8000$	Maximum interacting rounds
I_s	1000	Maximum rounds for transmitting phase
N_h	128	Hidden layer size
N_b	64	Batch size
c	15	Number of candidates (including the target)
lr	$\geq 10^{-5}, \leq 10^{-3}$	Learning rate

Table 3: Value of hyper-parameters.

APPENDIX B: TOPOLOGICAL SIMILARITY - A TOY EXAMPLE

Group	Compsitional (8)	low-ρ (16)	Other (232)
Language Examples	*blue box = aa* *red box = ba* *blue circle = ab* *red circle = bb*	*blue box = aa* *red box = bb* *blue circle = ab* *red circle = ba*	*blue box = aa* *red box = bb* *blue circle = aa* *red circle = bb*
ρ	1	0.5	0.1$\sim$ 0.7

Table 4: Different groups of language and their topological similarity.

To better understand how topological similarity can measure the compositionality of one language, and to provide some intuition on how mapping functions of languages corresponding to different values of ρ would be like, we extend here the toy example mentioned in section 2. In this example, the object space are $\mathcal{X} = \{\text{blue box, blue circle, red box, red circle}\}$ and message space $\mathcal{M} = \{aa, ab, ba, bb\}$. Any set of mappings from four distinct objects to four messages forms a language (a language may assign the same message to different object). Hence, we have 256 possible languages in this toy example, but only some of them can unambiguously describe the four possible objects: in this example, there is 24. Following the principles provided in (Kirby et al., 2015), we divide these 24 languages into two groups: compositional languages and holistic languages. We label the remaining 232 languages as 'Other', as illustrated in Table 4. The compositional languages, exhibit systematic structure when forming messages, while the holistic languages do not. In the compositional language in Table 4, m_1 and m_2 stands for the color and shape, respectively. We can hence use $S \rightarrow XY$, and X : *blue* $\rightarrow b; X$: *red* $:\rightarrow b; Y$: *box* $\rightarrow a; X$: *circle* $:\rightarrow b$ to represent such a language. But in holistic or other languages, such a structure may not exist.

Note that the number of compositional languages is usually smaller than that of holistic languages. When the neural agent is randomly initialized, all possible mappings (i.e., languages) in the searching space follow a uniform distribution. Hence the initial probability of specific groups of languages can be calculated using the ratio of such language types to all possible language types. Using permutation and combination, we can calculate the numbers of unambiguous language, compositional language and holistic language as:

$$\#\text{ unambiguous languages} = \frac{(|V|^{N_L})!}{(|V|^{N_L} - N_v^{N_a})!} \tag{6}$$

$$\#\text{ compositional languages} = N_a! \cdot \left(\frac{|V|!}{(|V| - N_v)!}\right)^{N_a} \tag{7}$$

$$\#\text{ holistic languages} = \#\text{ unambiguous languages} - \#\text{ compositional languages} \tag{8}$$

From the above equations, we can see that when N_L and $|V|$ increase, the gap between the number of compositional languages and holistic languages will become larger, which means that it is harder for us to select a compositional language when choosing at random. That is why the expected topological similarity of the emergent language may increase when smaller N_L and $|V|$ are applied, as illustrated in (Lazaridou et al., 2018; Cogswell et al., 2019).

Besides their initial probabilities, another key difference between these two types of languages can be illustrated using topological similarity. As the language studied in this paper is defined as a mapping function from a meaning (i.e., the object) to a message, a compositional mapping must ensure that the meaning of a symbol is a function of the meaning of its parts. In other words, the compositional language is neighborhood related: nearby meanings tend to map to nearby signals, because nearby meanings usually share similar attributes and hence are likely to share similar message symbols (Brighton & Kirby, 2006).

APPENDIX C: MORE ON THE LEARNING SPEED ADVANTAGE

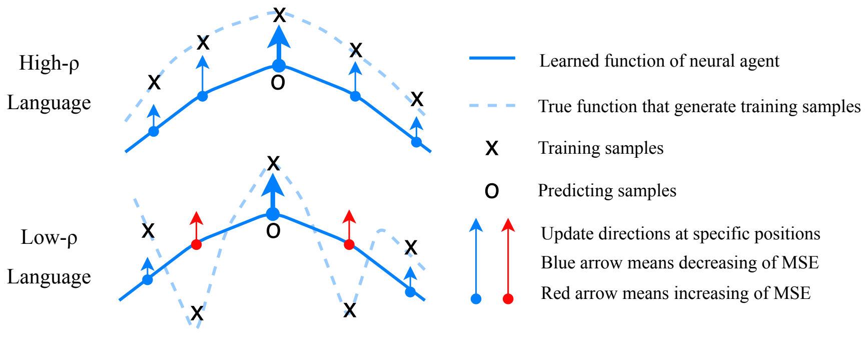

High-ρ Language
Low-ρ Language
X
O
Learned function of neural agent
True function that generate training samples
Training samples
Predicting samples
Update directions at specific positions
Blue arrow means decreasing of MSE
Red arrow means increasing of MSE

Figure 7: Illustration of learning a high-ρ language and low-ρ language.

Following the intuition that a language with higher ρ tends to be smoother and to have fewer inflection points than one with lower ρ, the learning speed advantage given by highly compositional languages can be illustrated by the example provided in Figure 7. In this example, language is considered to be a one-dimensional mapping function, which is represented by the dotted lines in the figure. The object-message pairs, which are represented by the cross marks, are the points that satisfy the mapping function. The solid line represents the mapping function of the learning agent. In the learning phase, the training data is randomly sampled from D_{i-1}. Suppose the target output (i.e., the third cross mark in each figure) is larger than the predicting output (i.e., the circle mark), the optimizer will update the parameters of the neural network following the direction of the gradient, as illustrated by the bold arrows in the figure. Such an update will also pull the neighbouring parts of the function up, as illustrated by the smaller arrows on the solid curve. The smoothness of high-ρ languages implies that the MSE of neighbouring positions will also be reduced by this update, while it would be increased in the case of a low-ρ language. Such a trend is represented by the blue and red arrows in the figure: the blue one means a decrease of the MSE at the specific position while the red one means the MSE is increasing. In other words, with high-ρ languages, an update corresponding to one data sample is likely to have a bigger effect, hence ensure a higher learning speed.

APPENDIX D: SUPPLEMENTARY EXPERIMENTS

Looking at the emergent languages

We first take a look at the number of different messages used by Alice in each generation, shown in Figure 8. There are 64 different objects in $\mathcal{X}$ and 64 possible messages in $\mathcal{M}$: hence, we expect that each possible message is allocated to a different object. In this figure, we can see that the speaking agents in all methods can use at least 56 different messages.After several generations, the resetting-both method performs slightly better than the others.

We then provide two examples of converged language (i.e., the language generated by Alice in the last generation) using the none-reset method and the resetting-both method in Table 5 and 6, respectively. In these examples, both languages can almost unambiguously represent all 64 different types of objects in $\mathcal{X}$, and hence they can help Alice and Bob to play the game successfully. However, the language generated using iterated learning has a clear compositional structure: the first position of the message represents different colors, and the second position represents the shape. Such a structure is quite similar to what humans do, e.g., combine an adjective and a noun to represent a complex concept.

Examining the posterior probability distributions

To better illustrate the posterior probability of emergent languages as a function of the corresponding value of ρ and the generation, we provide the 3D views of $P(\rho(\mathcal{L})|D_{i-1}, R_i)$ in 80 generations in Figure 10 and 11. The heat-map provided in Figure 9 can be considered as the top views of these 3D illustrations. In these two figures, the x-axis and y-axis represent the index of generation and the topological similarity, and the z-axis represents the probability of languages with a specific value of ρ, in a specific generation. To make the figures easier to read, we smooth the distribution of ρ in each generation using linear interpolation (Boyd & Vandenberghe, 2004).

Figure 12-(a) and (b) compare the posterior distributions at some typical generations, which can also be considered as the side views of the 3D illustration from the direction of x-axis. In these figures, we find that the initial distribution of ρ is not flat. That is because even the prior probability for each language is uniform, the amounts of languages with extremely high ρ and low ρ only occupy a small portion among all possible languages, as stated in (Brighton, 2002). Hence the initial probability of $\rho(\mathcal{L})$ is no longer uniform and has a bell shape which is similar to the Gaussian distribution. One new trend provided by these figures is that, in the none-reset case, the width of the curves in different generations do not change much, while in the resetting-both case, the width of the curves will gradually decrease (i.e., becomes more peaky). Such a trends means when iterated learning is applied, language tend to converge to some high-ρ types.

Figure 13-(a) and (b) track the ratio of languages with different values of ρ, which can also be considered as the side views of the 3D illustration from the direction of y-axis. In these figures, we divide all possible languages into five groups based on their topological similarity, i.e., languages with $\rho \leq 0.2$, $0.2 < \rho \leq 0.4$, $0.4 < \rho \leq 0.6$, $0.6 < \rho \leq 0.8$, and $0.8 < \rho$. We plot the ratio of these five different groups of languages at the end of each generation. From Figure 13-(a), we can see that the high-ρ language, which is represented by the bold curve, always occupy a small portion. The topological similarity of the dominant languages are $\rho < 0.4$. However, in the resetting-both case, as illustrated in Figure 13-(b), the portion of high-ρ language will increase significantly, which further verifies that the iterated learning can gradually make the high-ρ language dominate in posterior.

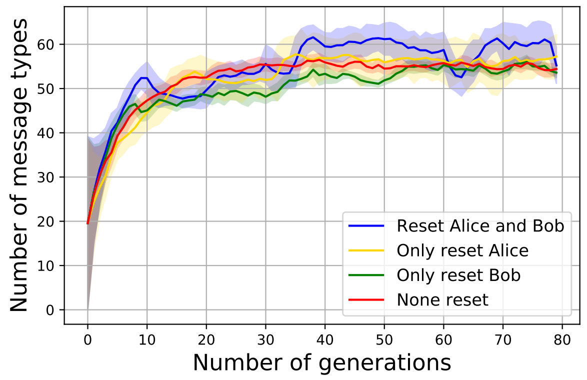


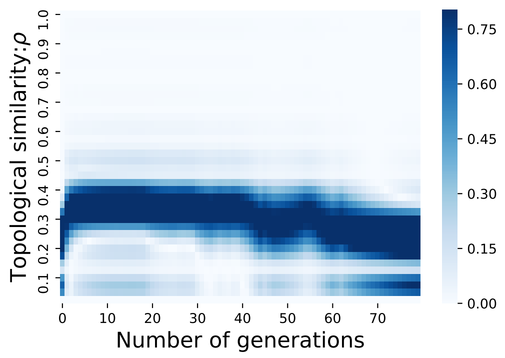
Figure 8: Message types of different settings.

	blue	green	cyan	brown	red	black	yellow	white
box	aa	fh	af	hh	cg	fc	ha	hf
circle	da	df	hb	db	fa	da	dh	fb
triangle	gc	ff	ge	gf	gg	fg	ge	he
square	ae	fb	be	bb	bg	fb	gb	ba
star	ad	fd	de	db	dg	fd	ce	hc
diamond	ac	dd	dc	db	dg	fd	dc	dd
pentagon	ad	fe	ef	bd	eg	fc	ee	ed
capsule	aa	dd	de	db	dg	gd	de	fh

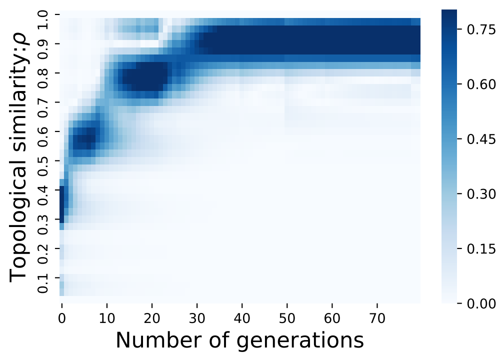
Table 5: Example of the converged language in none-reset case $\rho = 0.23$

	blue	green	cyan	brown	red	black	yellow	white
box	aa	ea	ba	ga	da	ca	ha	fa
circle	ab	eb	bb	gb	db	cb	hb	fb
triangle	ae	**eb**	be	ge	de	ce	he	fe
square	af	ef	bf	gf	df	cf	hf	ff
star	ac	ec	bc	gc	dc	cc	**dh**	fc
diamond	ad	ed	bd	gd	dd	cd	hd	fd
pentagon	ag	eg	bg	gg	dg	cg	hg	fg
capsule	ah	eh	bh	gh	**hc**	ch	hh	fh

Table 6: Example of the converged language in resetting-both case $\rho = 0.93$



(a) The none-reset case.



(b) The resetting-both case.

Figure 9: Distribution of $P(\rho(\mathcal{L})|D_{i-1}, R(G_i))$ through 80 generations. Values of ρ are divided into ten groups. The distribution of ρ in each generation is smoothed using linear interpolation.

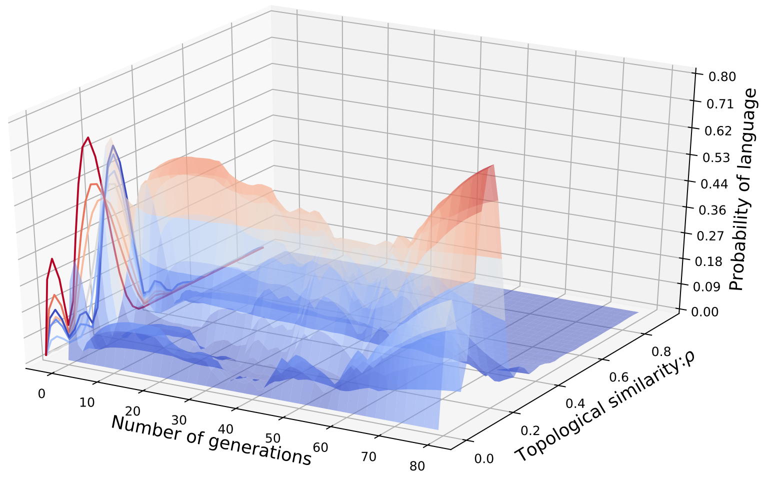


Figure 10: Language evolution of none-reset case in a 3D illustration.

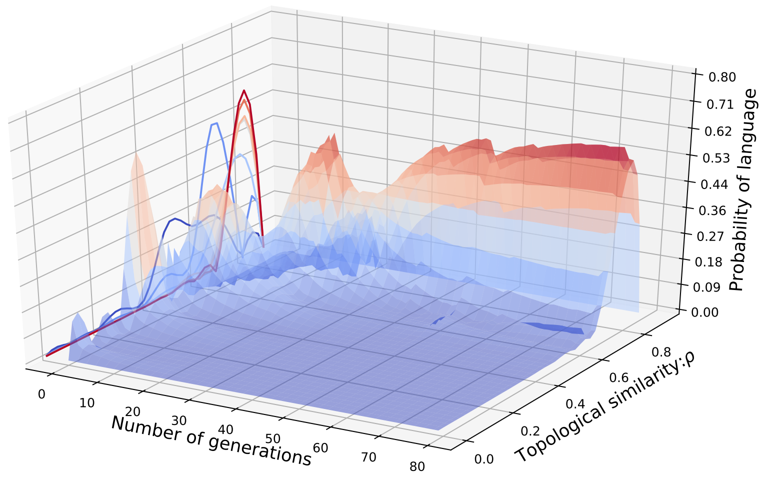
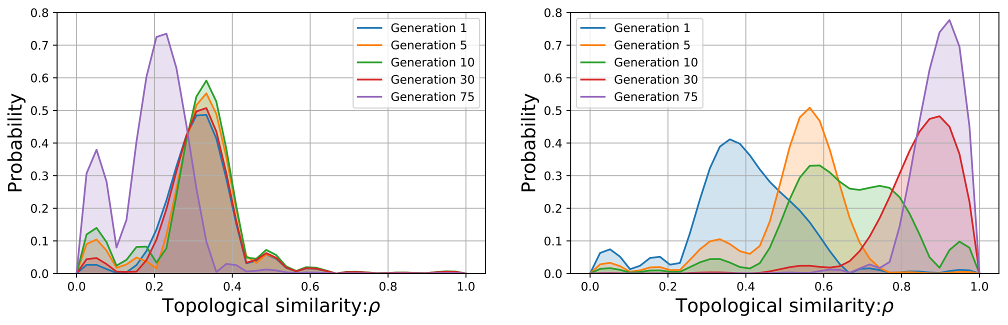


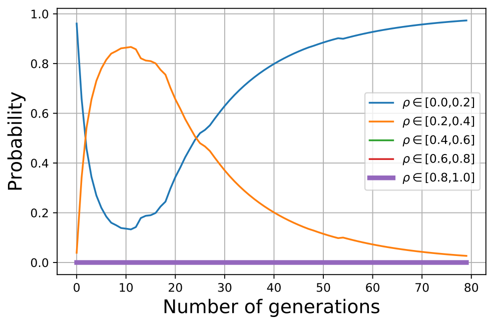
Figure 11: Language evolution of resetting-both case in a 3D illustration.



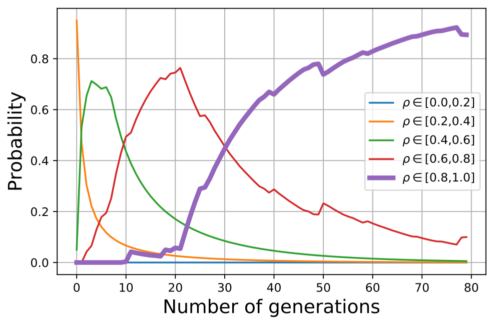
(a) None-reset case.

(b) Resetting-both case.

Figure 12: Distribution of $\rho(\mathcal{L})$ at different generations.



(a) None-reset case.



(b) Resetting-both case.

Figure 13: Evolution of language with different values of ρ.