Co-evolution of language and agents in referential games

Gautier Dagan
University of Amsterdam
gautierdagan@gmail.com

Dieuwke Hupkes
Facebook AI Research
dieuwkehupkes@fb.com

Elia Bruni
University of Osnabrück
elia.bruni@gmail.com

Abstract

Referential games offer a grounded learning environment for neural agents which accounts for the fact that language is functionally used to communicate. However, they do not take into account a second constraint considered to be fundamental for the shape of human language: that it must be learnable by new language learners.

Cogswell et al. (2019) introduced *cultural transmission* within referential games through a changing population of agents to constrain the emerging language to be learnable. However, the resulting languages remain inherently biased by the agents' underlying capabilities.

In this work, we introduce Language Transmission Simulator to model both cultural and architectural evolution in a population of agents. As our core contribution, we empirically show that the optimal situation is to take into account also the learning biases of the language learners and thus let language and agents *co-evolve*. When we allow the agent population to evolve through architectural evolution, we achieve across the board improvements on all considered metrics and surpass the gains made with *cultural transmission*. These results stress the importance of studying the underlying agent architecture and pave the way to investigate the *co-evolution* of language and agent in language emergence studies.

1 Introduction

Human languages show a remarkable degree of structure and complexity. In the evolution of this complex structure, several different intertwined *pressures* are assumed to have played a role. The first of these pressures concerns the function of language: as language is to communicate, it should allow effective communication between proficient language users (e.g. Smith and Kirby, 2012). This pressure is strongly intertwined with the nature of the proficient user: what features of language allow effective communication depends on the abilities of the user to use the language.

A second pressure on the shape of human language stems from the fact that language must be *learnable*. Unlike animal languages, which are taken to be mostly innate, human languages must be re-acquired by each individual (Pinker and Bloom, 1990; Hurford, 1998). A language can only survive if it can successfully be transmitted to a next generation of learners. In the field of language evolution, this transmission process is referred to as *cultural transmission*, while the process of change that occurs as a consequence is called *cultural evolution*.[1] Like the pressures arising from the function of language, the way that cultural evolution shapes language also depends on the language users: what is learnable depends on the inductive biases of the learner.

Computationally, the emergence of language can be studied through simulation with artificial agents and by investigating the resulting languages for structure, level of compositionality, and morphosyntactic properties (Kirby, 2001; Kirby and Hurford, 2002). Originally based on logic and symbolic representations (Kirby, 2001; Christiansen and Kirby, 2003), with the advent of modern deep learning methods, there has been a renewed interest in simulating the emergence of language through neural network agents (i.a. Lazaridou et al., 2017;

[1] The importance of cultural evolution for the emergence of structure is supported by a number of artificial language learning studies (e.g. Saldana et al., 2018) and computational studies using the Iterated Learning paradigm, in which agents learn a language by observing the output produced by another agent from the previous 'generation' (e.g. Kalish et al., 2007; Kirby et al., 2008, 2015).

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2993–3004
April 19 - 23, 2021.

Havrylov and Titov, 2017). Such work typically involves the use of *referential games* (Lewis, 1969), in which two or more agents have to emerge a language to obtain a shared reward.

These studies are motivated by the first pressure: language as a tool for effective communication. However, they fail to consider the second pressure: language must be learnable by new agents. They also fail to study the impact of the learning biases of the artificial agents themselves, which underlies both pressures. In a recent study, Cogswell et al. (2019) proposed a method to include cultural evolution in a language emergence game. Their approach is more naturally aligned with pressures in humans language evolution than single agent referential games (see e.g. Wray and Grace, 2007), but fails to account for the fact that cultural evolution and the learning biases of the artificial agents are two sides of the same coin: what language is learnable depends on the learning biases of the learner.

In this paper, we will therefore integrate the three components described above – communication, cultural evolution and learning biases – and setup a framework in which their interaction can be studied. This framework, which we refer to with the term Language Transmission Simulator, consists of *a referential game*, played by *a changing population of agents* – simulating cultural evolution – which are subject to *architectural evolution* – simulating the learning biases of the learners and allowing them to *co-evolve* with the language.

Our contributions are three-fold fold:

- We introduce the *Language Transmission Simulator*, that allows to model both cultural and architectural evolution in a population of agents;
- We collect a large number of tests from previous work and combine them into an extensive test suits for language emergence games;
- We demonstrate that emerging languages benefit from including cultural transmission *as well as* architectural evolution, but the best results are achieved when languages and agents can co-evolve.

2 Related Work

Much work has been done on the emergence of language in artificial agents and investigating its subsequent structure, compositionality and morphosyntax (Kirby, 2001; Kirby and Hurford, 2002). Originally, such work was based on logic and symbolic representations (Kirby, 2001; Christiansen and Kirby, 2003), but with the advent of modern deep learning (LeCun et al., 2015), there has been a renewed interest in simulating the emergence of language through neural network agents (i.a. Foerster et al., 2016; Kottur et al., 2017; Choi et al., 2018; Lazaridou et al., 2017; Havrylov and Titov, 2017; Mordatch and Abbeel, 2018). In the exploration of language emergence, different training approaches and tasks have been proposed to encourage agents to learn and develop communication. In a typical setup, two players aim to develop a communication protocol in which one agent must communicate information it has access to (typically an image), while the other must guess it out of a line-up (Evtimova et al., 2018; Lazaridou et al., 2017).

Kottur et al. (2017) show that 'natural' language does not arise naturally in these communication games and it has to be incentivised by imposing specific restrictions on games and agents. Havrylov and Titov (2017) first demonstrated that using straight-through estimators were more effective than reinforcement learning in a collaborative task, and that optimizing rewards can lead to structured protocols (i.e. strings of symbols) to be induced from scratch. Mordatch and Abbeel (2018) find that syntactic structure emerges in the stream of symbol uttered by agents, where symbols and syntax can be mapped to specific meanings or instructions. Choi et al. (2018) use qualitative analysis, visualization and a zero-shot test, to show that a language with compositional properties can emerge from environmental pressures.

Chaabouni et al. (2019) find that emerged languages, unlike human languages, do not naturally prefer non-redundant encodings. Chaabouni et al. (2020) further find that while generalization capabilities can be found in the languages, compositionality itself does not arise from simple generalization pressures. (Rodríguez Luna et al., 2020) encourage desirable properties of human languages, such as compositionality, to emerge through well-crafted auxiliary pressures. Finally, Harding Graesser et al. (2019) demonstrate with experiments on populations of agents that language can evolve and simplify through the interactions of different communities.

Cogswell et al. (2019) build upon the emergent language research by introducing cultural trans-

mission as a pressure in referential games. They use a pool of agents with a resetting mechanism and show that this further encourages the emerging language to display compositional properties and structure allowing it to generalize better. Pairing agents with one another in a larger population setting introduces cultural evolution, but it is the pressure introduced by the partial resetting which forces remaining agents to emerge a language that is quickly learnable by a new agent. While Cogswell et al. (2019) is the most related work to ours, an important difference is that they focus on cultural evolution only, without taking into account the learning biases of the agents via modelling architectural evolution.

3 Approach

In this paper, we introduce architectural evolution in language emergence games and study the interaction between cultural and architectural evolution with a range of different metrics. Below, we first give a definition of the referential game we consider (Subsection 3.1). We then briefly explain our Language Transmission Simulator (Section 3.2 and how we model cultural and architectural evolution within it (Section 3.3 and 3.4 respectively).

3.1 Referential games

We study language emergence in a referential game inspired by the signalling games proposed by Lewis (1969). In this game, one agent (the *sender*) observes an image and generates a discrete message. The other agent (the *receiver*) uses the message to select the right image from a set of images containing both the sender image and several distractor images. Since the information shown to the sender agent is crucial to the receivers success, this setup urges the two agents to come up with a communication protocol that conveys the right information.

Formally, our *referential* game is similar to Havrylov and Titov (2017):

1. The meaning space of the game consists of a collection D of K images $\{d_0, d_1, ..., d_K\}$, represented by z-dimensional feature vectors.
2. In each round i of the game, a target item d_i is randomly sampled from D, along with a set C of n distractor items.
3. The sender agent s of the game, parametrised by a neural network, is given item d_i, and generates a discrete message m_i from a vocabulary V. The message is capped to a max message length of L.
4. The receiver agent r, also parametrised by a neural network, receives message m_i and uses it to identify d_i in the union of d_i and C.

We use $z = 512$, and $n = 3$ and train agents with Gumbel-Softmax (Jang et al., 2017a) based on task-success.

3.2 Language Transmission Simulator

In Language Transmission Simulator, depicted in Figure 1, we simulate a population of communicating agents. In every training iteration, two random agents are sampled to play the game. This forces the agents to adopt a simpler language naturally: to succeed they must be able to communicate or understand all opposing agents. In our setup, agents are either sender or receiver, they do not switch roles during their lifetime.

3.3 Cultural evolution in referential games

Following Cogswell et al. (2019), we simulate cultural evolution by periodically replacing agents in the population with newly initialised agents. Cultural evolution is implicitly modelled in this setup, as new agents have to learn to communicate with agents that already master the task. Following Cogswell et al. (2019), we experiment with three different methods to select the agents that are replaced: randomly (no selection pressure), replacing the oldest agents or replacing the agents with the lowest fitness (as defined in Section 3.5). We call these setups `cu-random`, `cu-age` and `cu-best`, respectively.

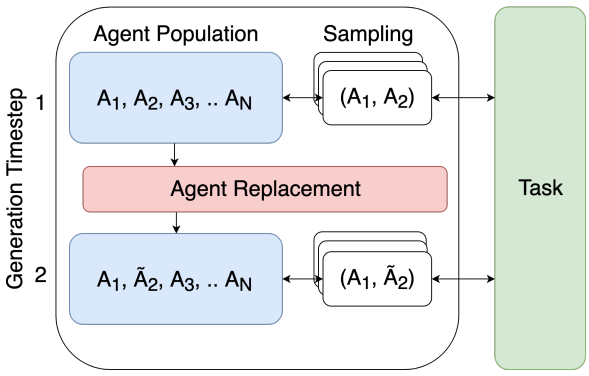

Figure 1: The Language Transmission Simulator: Agent pairs are randomly sampled from each population and trained. After l training steps, a portion α of the population is culled.

3.4 Architectural evolution in referential games

To model architectural evolution, rather than periodically replacing agents with randomly initialised new agents, we instead mutate the most successful agents and replace the worst agents with variations

of the best agents, as outlined in Section 3.4.2. Note that cultural evolution is still implicitly modelled in this setup, as new agents still have to learn to communicate with older agents. Therefore, we call this setup with the term `co-evolution`.

3.4.1 Culling

We refer to the selection process and subsequent mutation or re-initialisation step as *culling*. In biology, culling is the process of artificially removing organisms from a group to promote certain characteristics, so, in this case, culling consists of removing a subset of the worst agents and replacing them with variations of the best architecture. The proportion of agents from each population selected to be mutated is determined by the culling rate α, where $\alpha \in [0, 1)$. The culling interval l defines the number of iterations between culling steps. A formalisation of the LTE can be found in appendix A.1.

3.4.2 Mutation Algorithm

Our mutation algorithm is an intentionally simple implementation of a Neural Architectural Search (NAS). NAS focuses on searching the architecture space of networks, unlike many traditional evolutionary techniques which often include parameter weights in their search space. We opted to use the DARTS (Differentiable Architecture Search) RNN search space defined by Liu et al. (2018), which has obtained state-of-the-art performance on benchmark natural language tasks (Li and Talwalkar, 2019).

The DARTS search space includes recurrent cells with up to N nodes, where each node $n_1, n_2, ..., n_N$ can take the output of any preceding nodes including n_0, which represents the cell's input. All potential connections are modulated by an activation function, which can be the identity function, Tanh, Sigmoid or ReLU. Following Liu et al. (2018) and Pham et al. (2018), we enhance each operation with a highway bypass (Zilly et al., 2016) and the average of all intermediate nodes is treated as the cell output.

To sample the initial model, we sample a random cell with a single node ($N = 1$). As this node must necessarily be connected to the input, the only variation stems from the possible activation functions applied to the output of n_1, resulting in four possible starting configurations. We set a node cap of $N = 8$. We mutate cells by randomly sampling an architecture which is one edit step away from the previous architecture. Edit steps are uniformly sampled from i) changing an incoming connection, ii) changing an output operation or iii) adding a new node; the mutation location is uniformly sample from all possible mutations.[2]

3.5 Fitness Criterion

The fitness criterion that we use in both the `cu-best` and `co-evolution` setup is based on task performance. However, rather than considering agents' performance right before the culling step, we consider the age of the youngest agent in the population (defined in terms of number of batches that it was trained) and for every agent compute their performance up until *when they had that age*. For any agent a_j in population $\mathbf{A}$ this is defined as:

$$\text{fitness}(a_j) = \frac{1}{\mathcal{T}_A} \sum_{t=0}^{\mathcal{T}_A} \mathcal{L}(a_j^t) \qquad (1)$$

where $\mathcal{T}_A = \min_{a \in A} \mathcal{T}(a)$ is the age $\mathcal{T}(a)$ of the youngest agent in the population, and $\mathcal{L}(a_j^t)$ is the loss of agent a_j at time step t. This fitness criterion is not biased towards older agents, that have seem already more data and have simply converged more. It is thus not only considering task performance but also the *speed* at which this performance is reached.

4 Experiments

We test the LTE framework on a compositionally defined image dataset, using a range of different selection mechanisms.

4.1 Dataset

In all our experiments, we use a modified version of the Shapes dataset (Andreas et al., 2015), which consists of 30 by 30 pixel images of 2D objects, characterised by shape (circle, square, triangle), colour (red, green, blue), and size (small, big). While every image has a unique symbolic description – consisting of the shape, colour and size of the object and its horizontal and vertical position in a 3x3 grid – one symbolic representation maps to multiple images, that differ in terms of exact pixels and object location. We use 80k, 8k, 40k images for train, validation and test sets, respectively. Some example images are depicted in Figure 2.

[2] For a formal description of the mutation process, we refer to Appendix A.2.

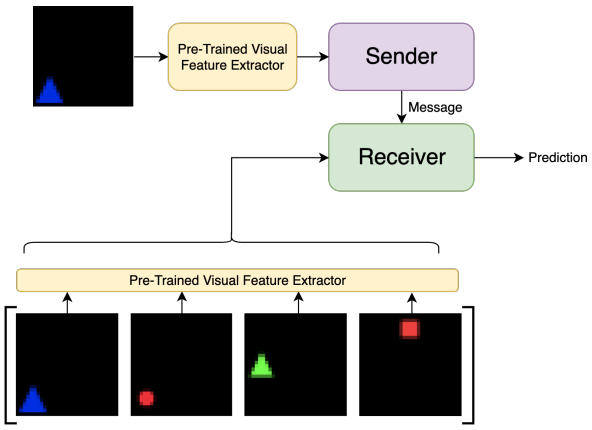

Figure 2: The modified Shapes task consists of showing an image the sender, and then letting the receiver deduce from the sender's message which image out of the target and k distractors is the correct one.

We pre-train a CNN feature extractor for the images in a two-agent setting of the task (see Appendix A.4 for more details).

4.2 Architecture and Training

For our co-evolution experiments, we use the DARTS search space as described above. For all cultural evolution approaches, we use an LSTM (Hochreiter and Schmidhuber, 1997) for both the sender and receiver architecture (see Appendix A.3 for more details). The sender and receiver models have a hidden size of 64 for the recurrent layer and an embedding layer of size 64. Further, we use a vocabulary size V of 4, with an additional bound token serving as the indicator for beginning and end-of-sequence. We limit the maximum length of a sentence L to 5. [3]

We back-propagate gradients through the discrete step outputs (message) of the sender by using the Straight-Through (ST) Gumbel-Softmax Estimator (Jang et al., 2017b). We run all experiments with a fixed temperature $\tau = 1.2$. We use the default Pytorch (Paszke et al., 2017) Adam (Kingma and Ba, 2015) optimiser with a learning rate of 0.001 and a batch-size of 1024. Note that the optimiser is reset for every batch.

For all multi-agent experiments we use a population size of 16 senders and 16 receivers. The culling rate α is set to 0.25 or four agents, and we cull (re-initialise or mutate) every $l = 5k$ iterations.

[3]The values for V and L were picked to provide a strong communication bottleneck to promote the emergence of structured and compressed languages, following the intuitions from Kottur et al. (2017) that natural language patterns do not emerge 'naturally'.

We run the experiments for a total of $I = 500k$ iterations, and evaluate the populations before each culling step.

4.3 Evaluation

We use an range of metrics to evaluate both the population of agents and the emerging languages.

Jaccard Similarity We measure the consistency of the emerged languages throughout the population using *Jaccard Similarity*, which is defined as the ratio between the size of the intersection and the union of two sets. We sample 200 messages per input image for each possible sender-receiver pair and average the Jaccard Similarity of the samples over the population. A high Jaccard Similarity between two messages is an indication that the same tokens are used in both messages.

Proportion of Unique Matches We compute how similar the messages that different agents emit for the same inputs by looking at all possible (sender, message) pairs for one input and assess whether they are the same. This metric is 1 when all agents always emit the same messages for the same inputs.

Number of Unique Messages We compute the average *number of unique messages* generated by each sender in the population. An intuitive reference point for this metric is the number of images with distinct symbolic representations. If agents generate more messages than expected by this reference point, this demonstrates that they use multiple messages for the images that are – from a task perspective – identical.

Topographic Similarity Topographic similarity, used in a similar context by Lazaridou et al. (2018), represents the similarity between the meaning space (defined by the symbolic representations) and the signal space (the messages sent by an agent). It is defined as the correlation between the distances between pairs in meaning space and the distances between the corresponding messages in the signal space. We compute the topographic similarity for an agent by sampling 5,000 pairs of symbolic inputs and corresponding messages and compute the Pearson's ρ correlation between the cosine similarity of the one-hot encoded symbolic input pairs and the cosine similarity of the one-hot encoded message pairs.

Average Population Convergence To estimate the speed of learning of the agents in the population, estimate the *average population convergence*. For each agent, at each point in time, this is defined as the agents average performance from the time it was born until it had the age of the current youngest agent in the population (analogous to the fitness criterion defined in Section 3.5). To get the average population convergence, we take we average those values for all agents in the population.

Average Agent Entropy We compute the average certainty of sender agents in their generation process by computing and averaging their *entropy* during generation.

5 Results

We now present a detailed comparison of our cultural and co-evolution setups. For each approach, we averaged over four random seeds, the error bars in all plots represent the standard deviation across these four runs. To analyse the evolution of both agents and languages, we consider the development of all previously outlined metrics over time. We then test the best converged languages and architectures in a single sender-receiver setup, to assess the impact of cultural and genetic evolution more independently. In these experiments, we compare also directly to a single sender-receiver baseline, which is impossible for most of the metrics we consider in this paper. Finally, we briefly consider the emerged architectures from a qualitative perspective.

5.1 Task performance

We first confirm that all setups in fact converge to a solution to the task. As can be seen in Figure 3, all populations converge to a (close to perfect) solution to the game. The `cu-age` approach slightly outperforms the other approaches, with a accuracy that surpasses the 95% accuracy mark. Note that, due to the ever changing population, the accuracy at any point in time is an average of both 'children' and 'adults', that communicate with different members of the population.

6 Analysis

In this section we analyse the resulting behaviour and success of agents in Language Transmission Simulator. We first use standard approaches such as average agent entropy and loss (convergence) to measure the success of agents with respect to their language and the task. Secondly, we use other metrics to analyse the emergent language itself in terms of consistency and diversity by using Jaccard Similarity, the proportion of unique matches, the number of unique messages, and the topographic similarity. Thirdly, we perform a qualitative analysis of the architecture that emerge from our Language Transmission Simulator. Finally, we design Frozen Experiments, in which we test the emerged languages and architectures in a 1v1 setting with a fresh agent. This allows us to compare and measure the the improvement gains made by the architecture and those made by the language which emerged. We show through these experiments that the co-evolution setting leads to a language that is both more successful and easier to learn for a given new agent.

Figure 3: Average Population Accuracy of final populations.

6.1 Agent behaviour

To assess the behaviour of the agents over time, we monitor their average message entropy convergence speed. As can be seen in Figure 4, the `co-evolution` setup results in the lowest average entropy scores, the messages that they assign to one particular image will thus have lower variation than in the other setups. Of the cultural evolution setups, the lowest entropy score is achieved in the `cu-best` setup.

Figure 5 shows the average population convergence over time. Also in this case, we observe a clear difference between cultural evolution only and co-evolution, with an immediately much lower convergence time for co-evolution and a slightly downward trending curve.

6.2 Language Analysis

To check the consistencies of languages within a population, we compare the Jaccard Similarity and the Average Proportion of Unique Matches, which we plot in Figure 6. This shows that, compared to cultural evolution only, not only are the messages

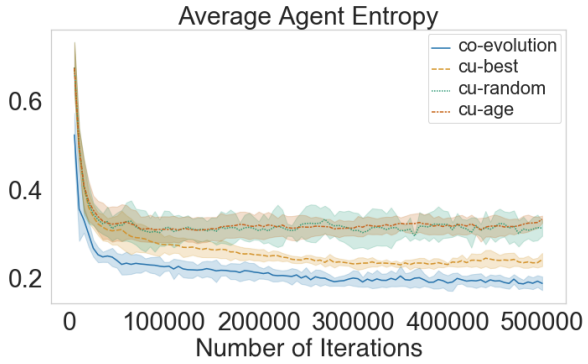

Figure 4: Average agent entropy over time.

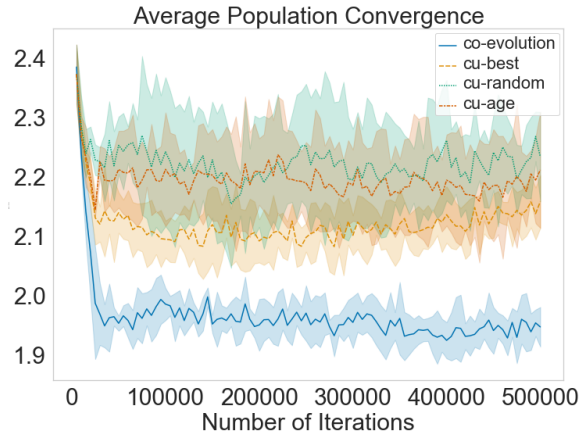

Figure 5: Average convergence for all cultural transmission modes and evolution.

in co-evolution more similar across agents (higher Jaccard Similarity), but also that agents are considerably more aligned with respect to the same inputs (less unique matches).

To assess the level of structure of the emerged languages, we plot the average Topographic Similarity and the Average Number of Unique Messages generated by all senders (Figure 7). The co-evolution condition again outperforms all cultural only conditions, with a simpler language (the number of the unique messages closer to the symbolic reference point) that is structurally more similar to the symbolic representation of the input (higher Topographical Similarity).

6.3 Architecture Analysis

In Figure 8 we show the co-evolution of an agent and a sample of its language during three selected iterations in the co-evolution setup. Strikingly, the best sender architecture does not evolve from its original form, which could point towards the limitations of of our search strategy and space. On

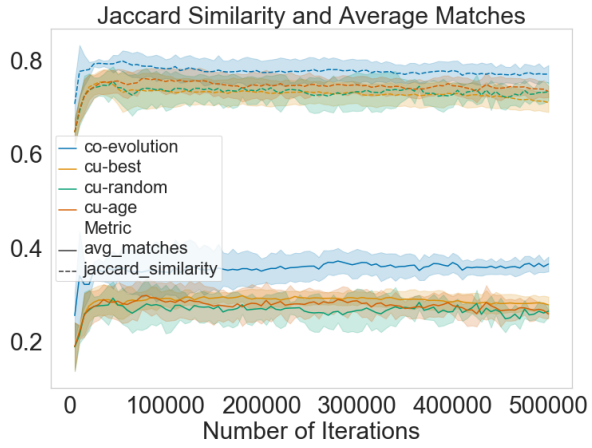

Figure 6: Average Jaccard Similarity and proportion of message matches for all cultural transmission modes and evolution

the contrary, the receiver goes through quite some evolution steps and converges into a significantly more complex architecture than its original form. We observe a unification of language throughout evolution in Figure 8, which is also supported by Figure 7. The population of senders starts out 11 different unique messages and ends with only two to describe the same input image. We will leave more detailed analysis of the evolved architectures for future work.

6.4 Frozen Experiments

With a series of experiments we test the a priori suitability of the evolved languages and agents for the task at hand, by monitoring the accuracy of new agents that are paired with converged agents and train them from scratch.

We focus, in particular, on training receivers with a frozen sender from different setups, which allows us to assess 1) whether cultural evolution made languages evolve to be more easily picked up by new agents 2) whether the genetic evolution made architectures converge more quickly when faced with this task. We compare the accuracy development of:

- An LSTM receiver trained with a frozen sender taken from `cu-best`;
- An evolved receiver trained with a frozen evolved sender.

For both these experiments, we compare with two baselines:

- The performance of a receiver agent trained from scratch along with a receiver agent

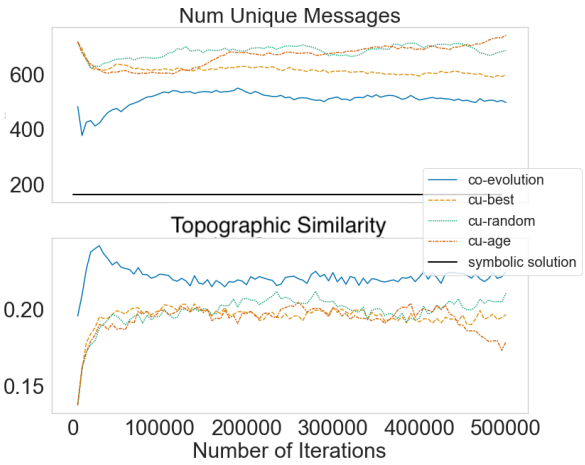

Figure 7: Average Number of Unique Messages and Topographic Similarity for all cultural evolution modes and co-evolution. For comparison, we also plot the number of unique messages for a symbolic solution that fully encodes all relevant features of the image (since we have three possible shapes and colours, two possible sizes, and a 3×3 grid of possible positions, this symbolic reference solution has $3 \times 3 \times 2 \times 9 = 162$ distinct messages.

that has either the `cu` architecture or the evolved `co` architecture (`cu-baseline` and `co-baseline`, respectively);

- The performance of an agent trained with an agent that is *pretrained* in the single agent setup, with either the `cu` architecture or an evolved architecture (`cu-baseline-pretrained` and `co-baseline-pretrained`).

Each experiment is run 10 times, keeping the same frozen agent. The results confirm cultural evolution contributes to the learnability and suitability of emerging languages: the `cu-best` accuracy (green line) converges substantially quicker and is substantially higher than the `cu-baseline-pretrained` accuracy (orange line). Selective pressure on the language appears to be important: the resulting languages are only easier to learn in the `cu-best` setup.[4] In addition, they show that the agents benefit also from the genetic evolution: the best accuracies are achieved in the co-evolution setup (red line). The difference between the `cu-baseline` (blue) and the `co-baseline` (brown) further shows that even if the evolved architectures are trained from scratch, they perform much better than a baseline model trained from scratch. The difference between the `co-baseline-pretrained` (only genetic evolution, purple line) and the co-evolution of agents and language line (red line) illustrates that genetic evolution alone is not enough: while a new evolved receiver certainly benefits from learning from a (from scratch) pretrained evolved sender, without the cultural transmission pressure, it's performance is still substantially below a receiver that learns from an evolved sender whose language was evolved as well.

[4] `cu-age` and `cu-random` are ommitted from the plot for clarity reasons.

7 Conclusion

In this paper, we introduced a language transmission bottleneck in a referential game, where new agents have to learn the language by playing with more experienced agents. To overcome such bottleneck, we enabled both the cultural evolution of language and the architectural evolution of agents, using a new Language Transmission Simulator. Using a battery of metrics, we monitored their respective impact on communication efficiency, degree of linguistic structure and intra-population language homogeneity. While we could find important differences in between cultural evolution strategies, it is when we included architectural evolution that agents scored best. In a second experiment, we paired new agents with evolved languages and agents and again confirmed that, while cultural evolution makes a language easier to learn, co-evolution leads to the best communication.

In future research, we would like to apply the Language Transmission Simulator on new, more complex tasks and further increase our understanding of the properties of the emerged languages and architectures. Recent research has also found that relaxing the vocabulary size V and sequence length L constraints can lead to greater syntactic structure in emergent languages (van der Wal et al., 2020). We thus hope to investigate further relaxation of hyper-parameters and other neuro-evolution techniques in future work.

Acknowledgments

We would like to thank Angeliki Lazaridou for her helpful discussions and feedback on previous iterations of this work.

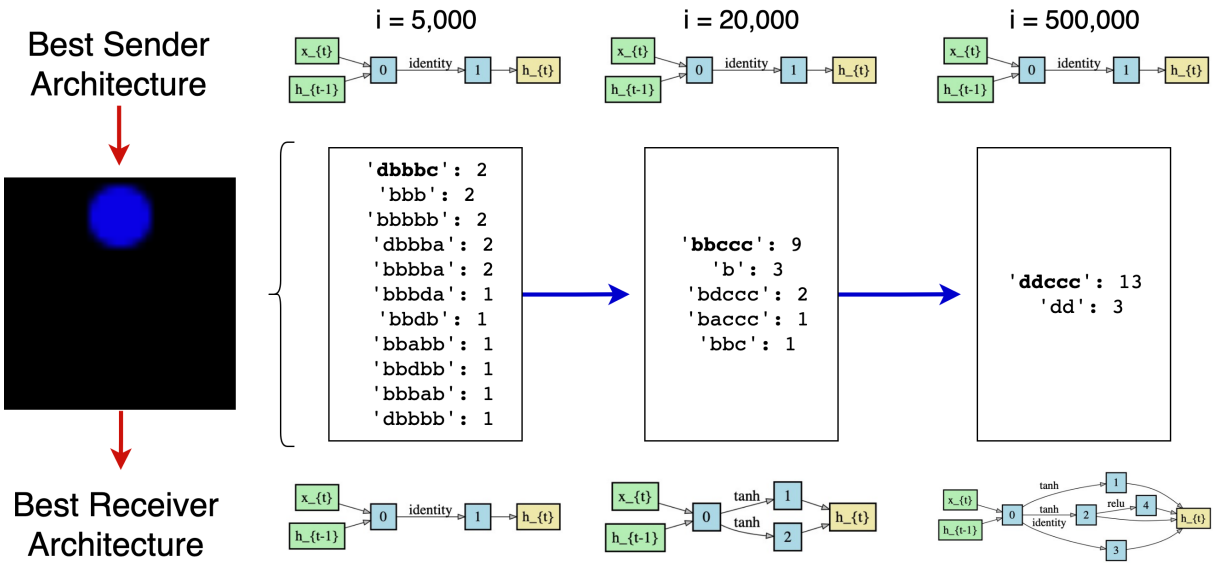

Figure 8: Evolution of the best sender and receiver architecture according to convergence, and the evolution of the population's message description of the same input through iterations. The bold messages represent the message outputted by the best sender whose architecture is pictured above. The count of each message represents the number of agents in the population which uttered this exact sequence.

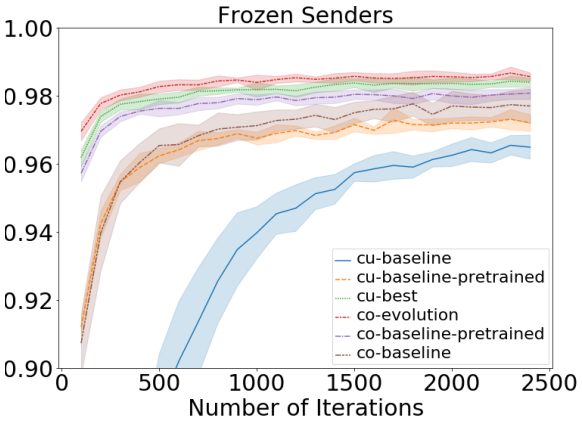

Figure 9: Receiver accuracies trained with different types of frozen senders.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2015. Deep Compositional Question Answering with Neural Module Networks.

Rahma Chaabouni, Eugene Kharitonov, Diane Bouchacourt, Emmanuel Dupoux, and Marco Baroni. 2020. Compositionality and generalization in emergent languages.

Rahma Chaabouni, Eugene Kharitonov, Alessandro Lazaric, Emmanuel Dupoux, and Marco Baroni. 2019. Word-order biases in deep-agent emergent communication. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5166–5175, Florence, Italy. Association for Computational Linguistics.

Edward Choi, Angeliki Lazaridou, and Nando de Freitas. 2018. Compositional obverter communication learning from raw visual input. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Morten H. Christiansen and Simon Kirby. 2003. Language evolution: consensus and controversies. *Trends in Cognitive Sciences*, 7(7):300–307.

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

Katrina Evtimova, Andrew Drozdov, Douwe Kiela, and Kyunghyun Cho. 2018. Emergent communication in a multi-modal, multi-step referential game. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. 2016. Learning to Communicate with Deep Multi-Agent Reinforcement Learning.

Laura Harding Graesser, Kyunghyun Cho, and Douwe Kiela. 2019. Emergent linguistic phenomena in multi-agent communication games. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3700–3710, Hong Kong, China. Association for Computational Linguistics.

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

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

James R Hurford. 1998. The evolution of language and languages. In *The evolution of culture*. Edinburgh University Press.

Eric Jang, Shixiang Gu, and Ben Poole. 2017a. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017b. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Micheal L. Kalish, Thomas L. Griffiths, and Stephan Lewandowsky. 2007. Iterated learning: Intergenerational knowledge transmission reveals inductive biase. *Psychonomic Bulletin & Review*, pages 288–294.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Simon Kirby. 2001. Spontaneous Evolution of Linguistic Structure-An Iterated Learning Model of the Emergence of Regularity and Irregularity. Technical Report 2.

Simon Kirby, Hannah Cornish, and Kenny Smith. 2008. Cumulative cultural evolution in the laboratory: an experimental approach to the origins of structure in human language. *Proceedings of the National Academy of Sciences of the United States of America*, 105(31):10681–6.

Simon Kirby and James R. Hurford. 2002. The Emergence of Linguistic Structure: An Overview of the Iterated Learning Model. In *Simulating the Evolution of Language*, pages 121–147. Springer London, London.

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

Satwik Kottur, José M. F. Moura, Stefan Lee, and Dhruv Batra. 2017. Natural language does not emerge 'naturally' in multi-agent dialog. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2962–2967.

Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. 2018. Emergence of linguistic communication from referential games with symbolic and pixel input. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. 2017. Multi-agent cooperation and the emergence of (natural) language. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

David Lewis. 1969. *Convention: A Philosophical Study*. Wiley-Blackwell.

Liam Li and Ameet Talwalkar. 2019. Random search and reproducibility for neural architecture search. *CoRR*, abs/1902.07638.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. DARTS: differentiable architecture search. *CoRR*, abs/1806.09055.

Igor Mordatch and Pieter Abbeel. 2018. Emergence of grounded compositional language in multi-agent populations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1495–1502.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. 2018. Efficient neural architecture search via parameter sharing. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 4092–4101.

Steven Pinker and Paul Bloom. 1990. Natural language and natural selection. *Behavioral and Brain Sciences*, 13(4):707–27.

Diana Rodríguez Luna, Edoardo Maria Ponti, Dieuwke Hupkes, and Elia Bruni. 2020. Internal and external pressures on language emergence: least effort, object constancy and frequency. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4428–4437, Online. Association for Computational Linguistics.

Carmen Saldana, Simon Kirby, Rob Truswell, and Kenny Smith. 2018. Compositional hierarchical structure evolves through cultural transmission: an experimental study.

Kenny Smith and Simon Kirby. 2012. *Compositionality and Linguistic Evolution*. Oxford University Press.

Oskar van der Wal, Silvan de Boer, Elia Bruni, and Dieuwke Hupkes. 2020. The grammar of emergent languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3339–3359, Online. Association for Computational Linguistics.

Alison Wray and George W. Grace. 2007. The consequences of talking to strangers: Evolutionary corollaries of socio-cultural influences on linguistic form. *Lingua*, 117(3):543–578.

Julian G. Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. 2016. Recurrent highway networks. *CoRR*, abs/1607.03474.

A Appendix

A.1 Language Transmission Engine

We formalise our Language Transmission process in the pseudo code shown in Algorithm 1. We select hyper-parameters l as the number of iterations or batches shown between culling steps, and I as the total number of iterations.

Algorithm 1 Language Transmission Engine

$S \leftarrow \{s_0, s_1 ..., s_N\}$
$R \leftarrow \{r_0, r_1 ..., r_N\}$
$i \leftarrow 1$
while $i \leq I$ **do**
  for batch b in D **do**
    Sample $\hat{s}$ from S
    Sample $\hat{r}$ from R
    $\mathbf{train}(\hat{s}, \hat{r}, b)$
    if $i \bmod l = 0$ **then**
      $\mathbf{cull}(S, R)$
    end if
    $i \leftarrow i + 1$
  end for
end while

A.2 Mutation Algorithms Pseudo-code

The genotype mutation is described in pseudo-code by algorithm 2, and takes as input a genotype containing nodes describing the cell. The genotype is mutated by either changing the input connection or primitive (output activation function) for a randomly sampled node n, or adding a new node altogether. See section 3.4.2 for explanations on the workings of the DARTS cell structure.

Algorithm 2 Genotype-level Mutation

procedure $\mathbf{mg}(genotype)$
  $g \leftarrow copy(genotype)$
  $a \leftarrow \mathcal{U}(1, 3)$
  $n \leftarrow \mathcal{U}(1, len(g))$
  if $a = 1$ **then**
    $p \leftarrow \mathcal{U}[ReLU, I, \tanh, \sigma]$
    $n.activation \leftarrow p$
  end if
  if $a = 2$ **then**
    $r \leftarrow \mathcal{U}(1, n)$
    $n.connection \leftarrow r$
  end if
  if $a = 3$ **then**
    $n' \leftarrow new\_node()$
    $p \leftarrow \mathcal{U}[ReLU, I, \tanh, \sigma]$
    $r \leftarrow \mathcal{U}(1, len(g))$
    $n'.activation \leftarrow p$
    $n'.connection \leftarrow r$
    $g.append(n')$
  end if
  return g
end procedure

Algorithm 3 Population-level Mutation

procedure $\mathbf{mutate}(\mathbf{P})$
  $p' \leftarrow \arg\min_{convergence}(P)$
  $\mathbf{p} \leftarrow \pi(P)$
  for p_i in $\mathbf{p}$ **do**
    $p_i.genotype \leftarrow \mathbf{mg}(p'.genotype)$
  end for
end procedure

In order to mutate a population $\mathbf{P}$ using π as a replacement policy, we use the process outlined in algorithm 3.

A.3 Agent Architecture

A.3.1 Sender Architecture

The sender architecture comprises of a linear layer input mapping the input feature size (512) to the hidden size. The image feature vector is therefore mapped to the same dimension as the RNN layer, where it is used as the initial hidden state. When training, for each step of the sender RNN we apply the cell and use the straight-through Gumbel-

Softmax trick to be able back-propagate gradients through the discrete message output. During evaluation however, we sample the categorical distribution at each step to produce each token in the sentence.

A.3.2 Receiver Architecture

The receiver architecture is simpler and takes as an input the message outputted by the sender and outputs a vector of input feature size (512). A single embedding matrix is used to encode the sender's message. During training the message is linearly transformed using the embedding matrix, while during the evaluation pass the discrete message outputs of the sender are used to map to the specific embedding dimensions. The embedded message is then passed to the RNN layer, and the final state of the RNN is linearly mapped back to the feature size. Doing so allows us to obtain a prediction for each image feature (distractors and true image), by comparing the alignment between the receiver output and the respective feature vectors.

A.4 Feature Extraction

In order to obtain image features, we pre-trained a convolutional model on the task using the raw image as input. Due to the input size requirements of the convolutional model, we resize the images linearly to be 128 by 128 (height, width) by 3 (RGB channels). We used early stopping conditions on the validation accuracy, an embedding size of 256, and hidden size of 512. The two agents are otherwise trained with the same parameters as other experiments: vocab size and max sentence length of 5, Adam optimizer with learning rate of 0.001.

For the visual module itself, we used a similar architecture to that in Choi et al. (2018) albeit smaller. We used a five-layer convolution network with 20 filters, and a kernel size and stride of 3 for all layers. For every convolutional layer, ReLU activation was applied on the output, after a Batch normalization step with no bias parameter. The linear layer which followed the convolutional layers had output dimensions of 512 and a ReLU activation function. This allows us to obtain image features of size 512, which we then used for all experiments.

A.5 Additional Figures and Analysis

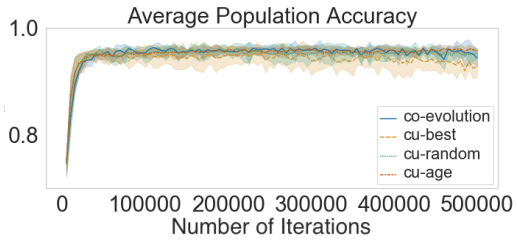

Figure 10: Average Population Accuracy for all Iterations