Language emergence in neural agents

Imen Ayadi **Julie Dessaint** **Matthieu Futeral** **Lucie Galland**

1 Introduction

Language emergence in neural agents is the task of building models to simulate language, in these models a speaker encodes a message while a listener decodes it. The purpose of this project is to investigate the properties of the encoded messages. Our work will be based on a standard model (Chaabouni et al., 2019) (a single block LSTM encoder-decoder) and the LazImpa model (Rita et al., 2020) which is similar to the previous model but penalizes the loss function differently in order to force the emergence of *efficient* encoded messages following the Zipf Law of Abbreviation (ZLA). In this project, we will first replicate LazImpa paper (Rita et al., 2020) experiments. Then we will work on different extensions: compositionality of language, attention mechanism into the listener network, disturb the speaker's message and trying new architectures for the listener and speaker.

2 Replication of experiments

In this section we will present the results we obtained when reproducing (Rita et al., 2020) experiments. The goal here is to observe the emergence of ZLA-compatible message in "LazImpa" communication agent. In the Lazimpa model, the speaker is made increasingly lazy, i.e., avoids long messages, and the listener impatient, i.e., seeks to guess the intended content as soon as possible. We will compare "LazImpa" agent to a standard communication agent as well as a Standard listener & Lazy speaker agent and an Impatient listener & Standard speaker agent. To compare the different agents, we will use the information metrics introduced in (Rita et al., 2020):

- $\mathbf{L_{type}}$, the mean message length.
- $\mathbf{L_{token}}$, average length of the messages weighted by their generation frequency.
- $\mathbf{L_{eff}}$, the mean number of informative symbols per message.
- $\rho_{\mathbf{inf}}$, fraction of informative symbols in the language.

To reproduce the experiment of the paper we have trained four different agents, experimenting with standard \lazy speaker and standard \impatient listener. The speaker and listener are modeled with one LSTM layer each. For all the experiments, the vocabulary size is set to 40, the maximum length of each message is set to 30 and the input space size is 100.

Table 1 presents the results we obtained for our four agents. We can clearly see that only the LazImpa model ensures the emergence of an efficient language. The mean length per message $\mathbf{L_{type}}$ is a lot smaller for the LaZimpa model than for the other ones. We can also see that the average length per messages weighted by their frequency $\mathbf{L_{token}}$ is smaller than $\mathbf{L_{type}}$ for the LazImpa agent which indicates the emergence of a language following the ZLA low where frequent messages are shorter than infrequent ones. Despite using a lot of symbols in their messages, the number of informative symbols $\mathbf{L_{eff}}$ is relatively small for the three other agents which demonstrates the inefficiency of these emerging languages. Finally, the ablation study shows that both the lazy speaker and impatient listener are needed to ensure the emergence of an efficient language.

	$\mathbf{L_{type}}$	$\mathbf{L_{token}}$	$\mathbf{L_{eff}}$	$\rho_{\mathbf{inf}}$
LazImpa	3.41	2.79	2.14	0.63
Standard Agent	28.89	29.03	3.75	0.12
Imp L.	29.71	24.41	2.01	0.07
Lazy S.	29.43	29.01	4.95	0.13

Table 1: Reproduction of (Rita et al., 2020) results. Efficiency and information analysis of emergent codes.

3 Additional experiments

3.1 Compositionality

Natural Languages follows the principle of compositionality which states that the meaning of a all sentence can be decomposed into the meaning of the sub-parts of the sentence. Then if the meaning of "a purple square" and "a blue rectangle" are known, one can understand "a purple rectangle". Modelling humans as Lazy and Impatient is common in cognitive sciences, therefore one can suppose that giving these properties to the speaker and the listener might help the principle of compositionality to emerge in the language created by the agents to solve the game. This hypothesis will be explored in this subsection.
At this effect, the experiment presented in the LazImpa paper was sightly modified. The set of considered vectors becomes $[\![0, n_{values} - 1]\!]^{n_{attributes}}$ where each dimension represents one attribute. In the following experiments, we choose $n_{values} = 10$ and $n_{attributes} = 2$. This set of vectors can be seen as a set composed of different shapes (square, circle...) of different color (blue, red...) for example. Each attribute is sampled according to a powerlaw, the one-hot vector representing each of the attributes are concatenated and given as input to the speaker.

The loss considered is the sum of the loss of the previous experiment, computed independently on each attribute.

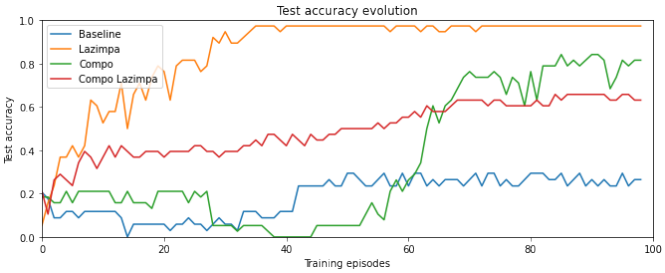


Figure 1: Test Accuracy on compositionality experiement

A subset of the possible combinations will be selected as the test set, the compositionality of the emerged language will be measured by the accuracy on this test set.

4 architectures will be tested :

- Baseline : A standard Speaker and a standard Listener
- LazImpa : A lazy speaker and an impatient Listener
- Compo : One strandard speaker and $n_{attributes}$ standard Listeners : each Listener is specialized in one attribute
- LazImpa Compo : One lazy speaker and $n_{attributes}$ impatient Listeners

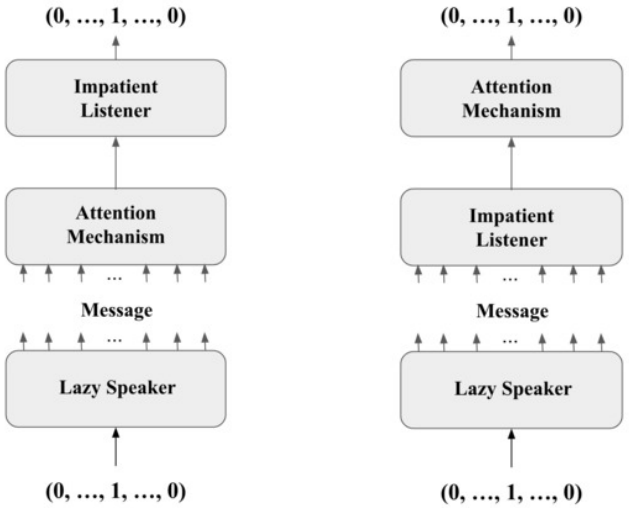
The "compo" architecture composed of one speaker and $n_{attributes}$ listeners is designed to facilitate the emergence of compositionality in the created language. Each Listener answers to a question on the original vector : 'Which shape ?' or 'Which color ?'.

The results shows that using lazy speaker and impatient listeners helps the emergence of the compositionality principle (see fig1). The Baseline is unable to solve the task both on the train and test set. The "Compo" architecture makes the training more difficult and the convergence longer. Indeed a Listener training on both attributes is learning more about the language than a Listener who cares only about one attribute. However it allows standard agents to solve this task. Nevertheless, the only architecture able to solve the task on both the train and test set is one lazy speaker and one impatient listener. Therefore it seems that using lazy/impatient agent allows the emergence of other principle of natural languages such as compositionality, however it would be interesting to test other sampling of the attribute since the powerlaw sampling might have favor the Lazimpa framework.

3.2 Attention Listener

Methodology

The purpose of this experiment is to introduce an attention mechanism (Bahdanau et al., 2015) into the Listener network in order to identify which tokens the listener focuses on and analyse how the network makes decisions, and particularly why it cannot decode the messages sometimes. Concretely, based on the LazImpa agents (Rita et al., 2020), two architectures were tested : the first one is a Lazy Speaker and an Impatient Listener with an attention mechanism directly applied to the word embedding of the message, the second one is a Lazy Speaker and an Impatient Listener with an attention mechanism applied to the output of the Listener. The models are illustrated in figure 2. More specifically, the setup was the same as the one mentioned in the original paper : there are 100 different inputs with 40 words in the vocabulary and the maximum length of the sentence is 30. In addition, both Lazy Speaker and Impatient Listener are modeled with one LSTM layer each.



(a) Attention LazImpa : first model

(b) Attention LazImpa : second model

Figure 2: Position of the attention mechanism in the experiments conducted

Quantitative Results

	Power.	Unif.
Lazy S. + Imp. L.	0.998	0.99
Lazy S. + Attn M. + Imp. L.	0.988	0.97
Lazy S. + Imp. L. + Attn M.	0.891	0.68

Table 2: Powerlaw and uniform scores over all the inputs for both attention experiments and LazImpa agents

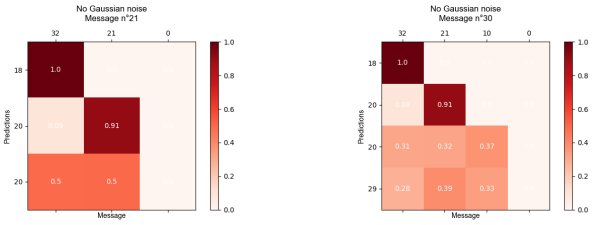
Table 2 shows that a drop of performance was observed in both experiments particularly if the attention mechanism is applied to the output of the Impatient Listener. We will then focus on the first model (figure 3a) where the attention mechanism is applied to the message of the Lazy Speaker which has only a slight drop of performance.

Analysis of attention scores

We are then able to identify on which tokens the Impatient Listener focuses on to predict the message index and to draw conclusions from those observations.

1. First of all, it is possible to notice that most messages associated with less frequent inputs are extensions of messages associated with more frequent inputs. Therefore, in case of less frequent inputs, the Listener is likely to have a quite balanced distributed attention

over the sequence whereas in case of more frequent inputs, the Listener is likely to strongly focus on specific words.



(a) Attention LazImpa: message 21 (b) Attention LazImpa: message 30

Figure 3: Distribution of the attention scores over different sequences

From the example of the figure 3, one can notice that the 30th message is composed of the same words as the 21st message plus one other word. The latter has a balanced attention score over the sequence (≈ 0.33 for all words) which partly explains the fact that the Listener predicts the 30th message and not the 21st message.

2. It is possible to identify group of words (tokens) with a more powerful meaning whereas some are not used at all.

   First of all, some of the words are very used whereas some are not all. Indeed, some words are not even used to build the message. In the following figure, it seems that the words in the vocabulary follow a frequency/rank Zipf's law.

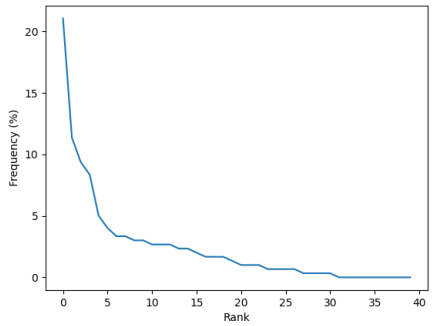


Figure 4: Rank vs frequency for the 40 words in the vocabulary

Then, it is interesting to analyze the average attention score of the words and to compare it with their frequencies. To perform it, we computed the Pearson correlation between the frequency of the words (or # of occurrences) and the average attention scores of each word and we got $\rho = -0.052$. Therefore, there is no correlation between the average attention score of a word and its number of occurrences although we thought that infrequent words would be more informative than frequent words. In addition, eight words out of the forty words in the vocabulary are not used at all.

3. **Positional encoding**: To measure the importance of the position of a word in the prediction, we computed the average relative attention score over the position in the sequence. In other words, for each input, we compared the attention score of a token at position k with the attention score if each token in the sequence were not informative. For example, if the length of the message is 3, the relative attention score of a token at position k is : attn score$(k) - 1/3$. Then, from table 3, it is possible to conclude that the position has not a significant impact in an emergent code because the average relative attention score is around 0 for each position except for the first position that seems to be slightly less informative than other positions.

k	N(k)	Avg. Relative Attn. score
Pos. 1	99	-0.11
Pos. 2	94	0.06
Pos. 3	64	0.05
Pos. 4	37	0.05
Pos. 5	5	-0.001
Pos. 6	0	0

Table 3: Average relative attention score over the position in the sequence. N(k) is the number of messages that have a symbol (different from EOS) at position k.

3.3 Disturb speaker's messages

Methodology

The main idea is to slightly disturb the messages in order to analyze how the predictions of the impatient listener change over the new sequence of tokens, how the attention scores are newly distributed over the new sequence and to what extent the speaker's messages are robust to the introduced noise. More specifically, a Gaussian noise was added to the output of the Lazy Speaker before feeding it to the classifier to output the next token of the message. Concretely,

$$h_{LazySpeaker} \longleftarrow h_{LazySpeaker} + \epsilon_\alpha$$

where, $\epsilon_\alpha \sim \mathcal{N}(0, \sigma_\alpha^2)$

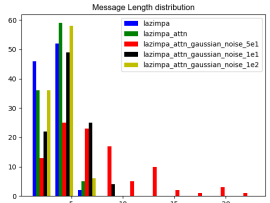
$$\sigma_\alpha^2 = \{1, \frac{1}{2}, \frac{1}{5}, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$$

To perform this experiment, we used the trained Attention LazImpa described in the previous section where the Attention Mechanism is applied directly to the Speaker's message.

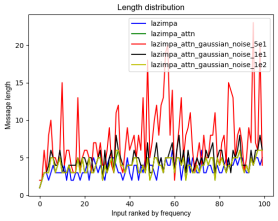
Analysis

As expected, the bigger is the standard deviation, the longer are the messages and the further the message length distribution is from Zipf's law. Nevertheless, It seems that the Attention LazImpa and the Attention LazImpa + gaussian noise 1e-2 follows Zipf's law.

The noise also impacts the communication between the speaker and the listener, when the noise is important (variance of 1 or 0.5), the predictions are severely affected by



(a) Histogram of the length of the messages for each model



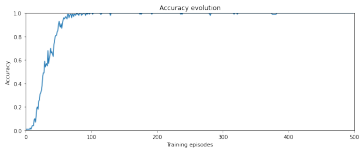
(b) Message length as a function of input frequency ranked

the noise and the listener has trouble decoding the messages. As you can observe in Figure 5a, the length of a message tends to vary a lot when the variance of the noise is important, yet examples show that these additional symbols don't convey information, in most cases the listener has already guessed the correct messages with the first few symbols. In fact, these longer messages can mislead the listener which correctly guess the message with the first few symbols but the last few symbols of the message mislead it. Finally it is pretty hard to find patterns in these attention maps, in some cases the listener will correctly decode the message even with a lot of changes and in other cases the listener will fail to decode the message even when the changes are pretty small. To gain more knowledge on how these perturbations affect the listener, we could perturb the message in some specific ways to study, for instance, the impact of modifying the first symbol of a message or the second one or adding more 'noisy' symbols in a message. We did not have the time to perform this extension but it could be done in the future to study how some specific alterations in the messages impact the listener.

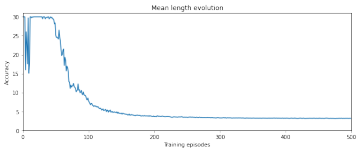
3.4 Replicate the experiments with new architectures

In (Chaabouni et al., 2019) and (Rita et al., 2020), the suggested architectures for both lazy speaker and impatient listener are basically a single-layer LSTM. The same architectures were used for standard agents; the only difference (apart from adding a penalization to the loss function) is that the impatient listener generates a prediction at each time step while the standard one consumes the entire message to yield a prediction. We tested other architectures for the Lazimpa agents and we observed the impact of this modification on the accuracy and the mean message length. In all experiments, the hyperparameters (except the architecture parameters) were kept as described in (Rita et al., 2020) and each time, we tuned the hyperparameters related of the network typology.

The first set of the tested architectures is the Gated Recurrent Unit (**GRU**), which belong as LSTM to the family of Recurrent Neural Network. GRUs are known to have a similar performance as LSTMs but, their main advantages over them is that GRUs have fewer parameters than LSTMs, as they lack an output gate. We implemented different experiments: (case 1)the speaker is a LSTM while the listener is a GRU, (case 2)the speaker is a GRU while the listener is a LSTM and (case 3)both agents are GRU. We notice that except case 1 , we reach the same accuracy as the baseline model with LSTM exclusively. In all the cases, the mean length converges to 5. Using GRU for the speaker made the accuracy evolution and the mean length evolution more stable (less fluctuations). The use of GRU for both agents (Figure 6) enabled us to speed up significantly the convergence compared with the use of LSTM, which was expected since GRU require less parameters to learn. Thus, in this set of experiments, **implementing the lazy speaker and the impatient listener with the Gated Recurrent Unit is the best choice in terms of convergence speed and stability.**



(a) Evolution of the accuracy with two GRU agents in the Lazimpa framework



(b) Evolution of the mean length with two GRU agents in the Lazimpa framework

Figure 6: GRU Architecture for both agents

More interestingly, transformers introduced in (Vaswani et al., 2017) were a good candidate to test. Unlike LSTM, the transformer does not imply any Recurrent Networks but uses instead a multi-headed self-attention. To inject the information of words' order, we used the sinusoidal positional embeddings. We tested two cases: the speaker is a transformer and the listener is a LSTM and vice versa. In the first experience, we notice that the evolution of the accuracy and the mean length i.e. the exploration step is much longer than in the case of LSTMs. Implementing the impatient listener as a transformer (after each time step, the listener decodes the message to keep the impatience of the agent) is even worser: after 500 epochs, we get a mean length of 8 words. Therefore, considering transformers was not suitable with the framework of Lazimpa. This is compatible with the attention analysis in section 3.2.

4 Discussion

In this project we have studied different aspect and properties of the emerging language between a speaker and a listener. First, we reproduced LazImpa's paper (Rita et al., 2020) experiment and studied how the lazy speaker and impatient listener enable the emergence of an efficient language following ZLA. Then, we studied an other property of natural language - compositionality - and studied if LazImpa communication agent help compositinality emergence. Then we took interest in adding an attention mechanism to the impatient listener in order to study which word the listener focuses on and to identify relevent patterns in the listener predictions. Then, we disturbed the message received by the listener with Gaussian Noise to see how this will impact the listener prediction. Finally, we tried different architectures for the listener and speaker and observed the impact on the accuracy and mean message length of the emerging language.

5 Contribution

I. Ayadi replicated the experiments with new architectures. J. Dessaint replicated the original experiments and worked on the *Disturb speaker's messages* experiment. M. Futeral worked on the Attention Listener agent and on the *Disturb speaker's messages* experiment. L. Galland worked on the Compositionality experiment.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Rahma Chaabouni, Eugene Kharitonov, Emmanuel Dupoux, and Marco Baroni. 2019. Anti-efficient encoding in emergent communication. *CoRR*, abs/1905.12561.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

.