“LazImpa”: *Lazy* and *Impa*tient neural agents learn to communicate efficiently
Supplementary Materials

Mathieu Rita[1] **Rahma Chaabouni**[1,2] **Emmanuel Dupoux**[1,2]
[1]Cognitive Machine Learning (ENS/EHESS/PSL Research University/CNRS/INRIA)
[2]Facebook AI Research
`mathieu.rita@polytechnique.edu, {rchaabouni,dpx}@fb.com`

A Appendix

A.1 Experimental settings

A.1.1 Input space

The input space $\mathcal{I}$ is composed of 1000 one-hot vectors. Each of them has to be communicated by Speaker to Listener. In order to fit the distribution of words in natural languages, the inputs are fed from a power-law distribution. Indeed, as demonstrated in Figure 1, distribution of words in natural languages follow power-laws with exponents k between -0.79 (Arabic) and -0.96 (Russian). In our experiment, we choose $k = -1$.

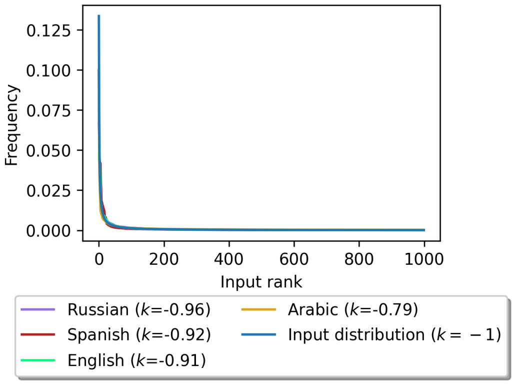


Figure 1: Comparison between the input distribution of our artificial environment and the distribution of the 1000 most frequent words in different natural languages (the coefficient k refers to the coefficient of the power-law for each language when fitted by a linear regression).

A.1.2 Agents

In all our experiments, we fix the architecture of the agents. Speaker is a 1-layer LSTM (Hochreiter and Schmidhuber, 1997) with a hidden size equal to 100. Listener is also a 1-layer LSTM with a hidden size equal to 600.

A.1.3 Optimization

For the training, we use the Adam optimizer (Kingma and Ba, 2014) with a learning rate equal to 0.001. We train the agents for 1500 epochs. During one episode, the system is fed with 100 batches of 512 inputs sampled with replacement from the power-law distribution. In addition, we enforce exploration with an entropy regularization coefficient equal to 2 (Williams and Peng, 1991).

To ensure the robustness of our results, we ran the experiments with 6 different random seeds. All the experiments have been successful, i.e. they reach an accuracy of 99%. This accuracy is weighted by the frequency of inputs. On average, more than 97.5% of inputs are well communicated.

A.1.4 Adaptive regularization coefficient

As defined in the main paper, the adaptive regularization coefficient is scheduled as a function of the accuracy in order to have the following two-step scheme:

- **Exploration step**: during the first part of the training (low accuracy), the regularization coefficient is almost null
- **Reduction step**: Once the communication becomes successful (high accuracy), we start introducing a regularization.

A fair equation to model this two-step scheme is:

$$\alpha(\text{accuracy}) = \frac{\text{accuracy}^{\beta_1}}{\beta_2} \qquad (1)$$

where $(\beta_1, \beta_2) \in \mathbb{R}^2$ is a new couple of hyperparameters. Intuitively, the two parameters allow to control (a) the threshold from which the regularization becomes effective (with β_1) and (b) the intensity of the regularization (with β_2). In our experiments, we introduce a late regularization choosing: $\beta_1 = 45$. We set $\beta_2 = 10$ in order to enables the system to reach an accuracy close to 1.
Note that other regularization scheduling can be applied. The only requirement is that the agents successfully communicate before the start of the reduction step.

A.2 Characterization of the emergent communication with Standard Agents

In this section, we report complements about the characterization of the emergent communication with Standard Agents.

A.2.1 Quick use of long messages

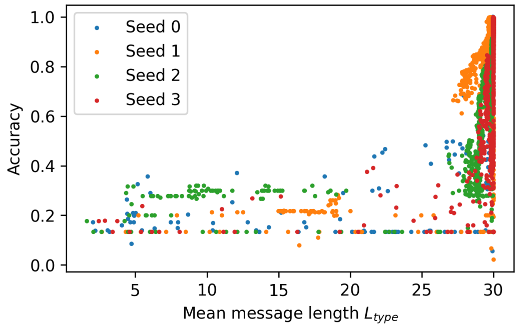


Figure 2: Accuracy as a function of the mean length for 4 different seeds. Each point represents a couple (accuracy, mean length).

To bring more insights about the length inefficiency observed in the main paper, we characterize each episode by the couple accuracy (i.e. the proportion of inputs correctly communicated by the agents weighted by the frequency of appearance) and mean length (i.e. the average length of the messages generated by the Speaker).

During the training time, we analyze how this couple evolves. The results with four randomly selected seeds are shown in Figure 2. As we can see, at the beginning of the learning process (low accuracies), both the mean length of the messages and the accuracy are quite low (the lowest accuracy value 0.13 corresponds to the good prediction of the most frequent input). Then, the mean message length is increasing without a strong effect on the accuracy. It is only when the agents start to use long messages (higher than 25 for a maximum length of 30) that the communication becomes successful. Therefore, we see that exploration of long messages seems key for the agents to reach high accuracies.

A.2.2 Efficient *informative* symbols

We analyze the statistical properties of the informative parts of the messages that emerge from Standard Agents. As defined in the main paper, we consider a symbol informative if it is used by Listener for the reconstruction. We remove all the non-informative symbols from the messages (i.e. positions k with $\Lambda_{k,\cdot} = 0$). In Figure 3, we plot the length of informative parts of messages associated to inputs ranked by frequency (average distribution over the different runs). We compare it to the average words length distribution of natural languages and to Optimal Coding. As we can see in the figure, even though Standard Agents produce an inefficient code (as seen in the main paper) the length statistic of the informative parts is close to Optimal Coding. Interestingly, we even note an emergent code more efficient than natural languages. In addition, even if no constraint is applied on informative parts, we observe that it follows ZLA.

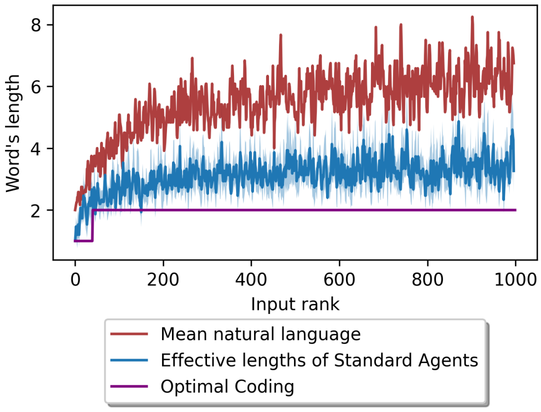


Figure 3: Average length distribution of informative parts in Standard Agents code compared to the mean words distribution of natural languages and Optimal Coding. The light blue interval shows 1 standard deviation. For readability, the natural language distribution have been smoothed with a sliding average of 3 consecutive lengths.

A.3 Comparing communication systems

A.3.1 Convergence

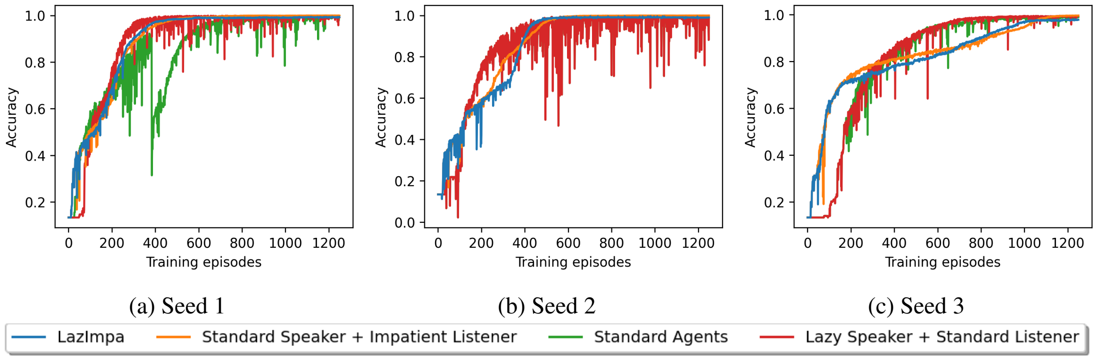
We check here the convergence and robustness of our introduced communication system, LazImpa. As a preliminary analysis, we compare the convergence results of: Standard Agents, (Standard Speaker + Impatient Listener), (Lazy Speaker + Standard Listener) and LazImpa. In Figure 4, we show the accuracy as a function of the training episodes for 3 randomly selected seeds. We see that the convergence dynamic is sensitive to the initialization but that in the end, the three systems converge.

Moreover, we observe a gain of stability for the systems with the Impatient Listener. Indeed, as shown in Figure 4, Standard Agents demonstrate a less smooth accuracy curve compared to both (Standard Speaker + Impatient Listener) and LazImpa. We quantify the stability by introducing a coefficient δ_{stab} that measures the local variations of the accuracy curves. Formally, we compute the mean square error between the original accuracy curve and the smoothed curve obtained by averaging 10 consecutive score values:

$$\delta_{stab} = \frac{1}{n}\sum_{i=1}^{n}(f(i) - \tilde{f}(i))^2 \qquad (2)$$

where n is the total number of episodes, $f(.)$ the accuracy curve (as a function of the number of episode), $\tilde{f}(i)$ the curve obtained by averaging $f(.)$ over with 11 consecutive episodes centered in i. The lower δ_{stab} is, the smoother the system is .

Results are reported in Table 1. δ_{stab} for systems with Impatient Listener are smaller than the one with Standard Listener confirming the stability of the former. It is important noticing that, contrary to (Chaabouni



(a) Seed 1 (b) Seed 2 (c) Seed 3

Figure 4: Evolution of the accuracy of the three systems for 3 randomly selected seeds.

	Standard Agents	Lazy Speaker + Standard Listener	Standard Speaker + Impatient Listener	LazImpa
δ_{stab}	$1.16 \pm 0.78 \times 10^{-3}$	$1.75 \pm 0.60 \times 10^{-3}$	$9.84 \pm 5.81 \times 10^{-5}$	$9.79 \pm 7.35 \times 10^{-5}$

Table 1: Average MSE between the original and smoothed accuracy curve

et al., 2019)'s setting where they managed to have more efficient languages at the cost of stable convergence, our new communicative system, on top of leading to efficient languages, has positive impact on the convergence.

A.3.2 Complement on randomization test

To be comparable with Ferrer i Cancho et al. (2013), we perform the randomization test with 10^{-5} permutations. In the reference article, for a threshold α they introduce two types of p-values:

- Left p-value: if left p-value $< \alpha$, the code is characterized by L_{token} significantly smaller than the average weighted message length of any random permutation, corresponding to our notion of *ZLA code*.
- Right p-value: if right p-value $< \alpha$, the code is characterized by L_{token} significantly higher than the average weighted message length of any random permutation, corresponding to our notion of *anti-ZLA code*.

In the main text, we only report the value of the ZLA significance score p_{ZLA} that is equivalent to Ferrer i Cancho et al. (2013)'s left p-value. However, when also considering right p-value (not shown here), we note for Standard Agents a value smaller than 10^{-5} asserting that the system shows a significantly anti-ZLA patterns.

A.4 Complements on LazImpa

A.4.1 minimal required length by Impatient Listener

Thanks to the incremental predictive mechanism of Impatient Listener, it is possible to analyze its intermediate guesses at each reading time. In particular, we are able to spot at which position Impatient Listener is first able to predict the correct output (we verify experimentally that, if Listener finds the correct output at position i, it always predicts the right output at position $j > i$). From these intermediate predictions, we define a distribution called 'minimal required length' of all the positions at which Impatient Listener is able to first predict the correct output (note that this distribution matches the distribution of the number of informative symbols by message).

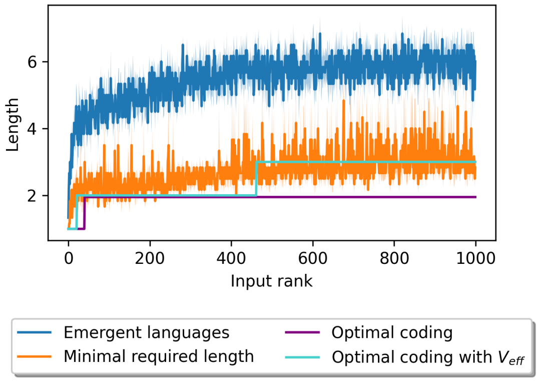


Figure 5: Comparison between the length distribution of the messages and the minimal required length for Impatient Listener to discriminate the messages. The blue curve shows average length distribution function of the inputs frequency ranks. The orange curve represents the average minimal required length by Impatient Listener to decode messages. The purple curve shows the Optimal Coding with the original vocabulary size. The red curve represents the Optimal Coding for the effective vocabulary size V_{eff}. Light intervals show 1 standard deviation.

We observe that Impatient Listener was often able to find the correct candidate before reading the `EOS` token. The resulting minimal length is presented in Figure 5 where we show the length distribution of the messages ranked by input frequency and the actual length required by the Impatient Listener to discriminate the messages. We see that the minimal required length by the Impatient Listener is slightly higher than the Optimal Coding. Interestingly, the difference can be partially explained by the use of a skewed distribution of the unigrams across the messages (the Optimal Coding relies on a uniform use of the symbols). Indeed, we compute an effective vocabulary size V_{eff}, solution of Equation 3:

$$-\sum_{i=1}^{V_{eff}} \frac{1}{V_{eff}} \log\left(\frac{1}{V_{eff}}\right) = \mathcal{H}(\mathcal{U}), \qquad (3)$$

where V_{eff} is the effective vocabulary size, and $\mathcal{H}(\mathcal{U})$ the entropy of the unigram distribution $\mathcal{U}$ in the emergent communication.

In other words, we search for V_{eff} for which the entropy of a uniform unigram distribution (the left side of Equation 3) is equal to emergent languages average unigram distribution (the right side of Equation 3).

We plot in Figure 5 a new Optimal Coding with V_{eff} (Optimal Coding with V_{eff}). The distribution 'minimal required length' almost fits the Optimal Coding with this vocabulary size. As shown in Table 2, the average mean length L_{type} of minimal required length is almost equal to L_{type} of Optimal Coding with V_{eff}.

A.4.2 LazImpa robustness to parameters assumptions

In this section, we analyze LazImpa robustness to parameters changes. In the main paper, we made two main assumptions:

1. Samples are drawn according to a powerlaw;
2. `voc_size` = 40 and `max_len` = 30.

In the main paper, we demonstrated that LazImpa is able to reach efficient performances with this set of assumptions. We now want to test whether the system is robust to changes of these parameters, i.e. is LazImpa able to produce efficient and successful codes when inputs are drawn uniformly and/or for different values of `voc_size` ? We report the results of all our experiments in Table 3. Curves associated to experiments with variations of vocabulary size are shown in Figure 6. All these results have been obtained by averaging the results over 3 different seeds by each set of parameters.

Effect of `voc_size` :

As we can observe in Figure 6, emergent codes still respects ZLA for the various tested values of vocabulary size. This is confirmed by the ZLA significance score p_{ZLA} stored in Table 3a. Additionally, we can see a correlation between the size of the vocabulary and the efficiency of the emergent code: the emergent code is more efficient for large sizes of vocabulary. Indeed, we observe that L_{type}, L_{token} and L_{eff} are increasing functions of the vocabulary size. This is expected as the number of messages of a given length increases with the vocabulary size. Thus, the set of 'short' messages is higher for a large vocabulary size. Naturally, the same trend is observed with Optimal Coding. Moreover, we note a decrease of ρ_{inf} as a function of `voc_size` for the LazImpa system, suggesting that the smaller the vocabulary size is the more noninformative positions are used.

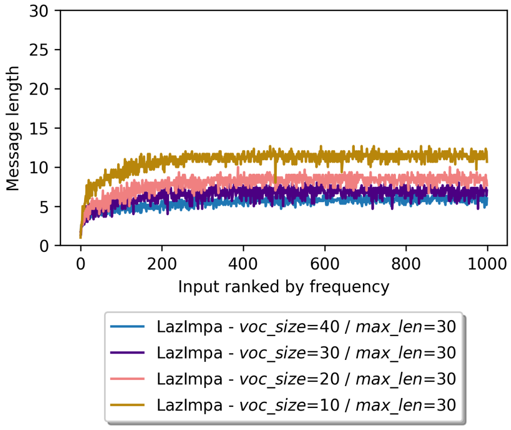


Figure 6: Comparison of LazImpa's average message length for different vocabulary sizes.

Effect of `max_len`: We can note in Table 3b that LazImpa is even closer to Optimal Coding when setting `max_len` = 20. L_{type}, L_{token} and L_{eff} are slightly smaller compared to experiments with `max_len` = 30. Thus, agents regularization seems to be easier when setting smaller values of `max_len`. Nevertheless, the results are very close. In particular, we can note that information density values ρ_{inf} are very similar suggesting that sub-optimality issues are independent of the parameter `max_len`. Note that we only explore two values of `max_len` in Table 3b because small and large values of `max_len` lead respectively to a small and large message space and thus optimization issues (H-parameters tuning is required to favor respectively exploration and exploitation).

Effect of input distribution: As we observe in Table 3c, LazImpa's performances are quite similar when dealing with inputs drawn from a uniform or a powerlaw distribution. In particular, with a uniform distribution, we observe a gain of efficiency for L_{type} and a loss of efficiency for L_{token} while L_{eff} is almost unchanged. All these results are expected. Equal L_{eff} means that Impatient Listener relies on the same number of symbols on average. In the main paper, we have shown that L_{eff} is mostly influenced by the entropy of the unigram distribution. Since, there is no change of `voc_size`, we do not expect major changes of entropy and thus no change for L_{eff}. Then, the difference of L_{token} and L_{type} is explained by the reduction step. For uniformly drawn inputs, the regularization is uniformly applied on the inputs ; for inputs drawn from a powerlaw, the regularization mostly focuses on the most frequent inputs because they have larger weights in the loss. Conse-

	Minimal required length	Opt. coding with V	Opt. coding with V_{eff}
L_{type}	2.74 ± 0.08	1.69	2.50

Table 2: Comparison of the average length L_{type} of different encoding. 'Opt. coding with V' to the Optimal Coding obtained with vocabulary V, 'Opt. coding with V_{eff}' to the Optimal Coding obtained with vocabulary V_{eff}. We also report standard deviation over all the experiments.

quently, we expect a lower L_{token} when experimenting with a powerlaw distribution, compared to the uniform setting, but a larger L_{type}. Eventually, we observe a significant gain of information density ρ_{inf} for LazImpa with a uniform distribution. This is mainly explained by ρ_{inf} computation that takes into account message lengths without involving their frequency.

As a remark, let's precise that we do not explore a larger set of non-uniform input distributions. In theory, the shape of the length distribution should not be impacted by the input distribution because the optimization problem is only dependent of the frequency ranks (mapping of the shortest messages to the most frequent inputs).

A.4.3 Statistical comparison between LazImpa and natural languages

Figure 7 shows the words length as a function of their frequency for both natural languages and the emergent language. This figure completes our comparison made in the main paper between LazImpa and natural languages where curves were smoothed. Here we show the raw natural languages distribution. The additional observation that we can make is that the variance of the words length is larger for the natural languages.

References

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

Ramon Ferrer i Cancho, Antoni Hernández-Fernández, David Lusseau, Govindasamy Agoramoorthy, Minna Hsu, and Stuart Semple. 2013. Compression as a universal principle of animal behavior. *Cognitive Science*, 37(8):1565–1578.

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

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Ronald Williams and Jing Peng. 1991. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3:241–.

`voc_size`	System	L_{type}	L_{token}	p_{ZLA}	L_{eff}	ρ_{inf}
40	LazImpa	5.49 ± 0.67	3.78 ± 0.34	$< 10^{-5}$*	2.67 ± 0.07	0.60 ± 0.07
	Optimal Coding	2.96	2.29	$< 10^{-5}$*	1.96	1
30	LazImpa	6.49 ± 1.20	4.14 ± 0.43	$< 10^{-5}$*	2.71 ± 0.22	0.53 ± 0.07
	Optimal Coding	3.09	2.35	$< 10^{-5}$*	2.09	1.
20	LazImpa	7.91 ± 0.71	4.80 ± 0.30	$< 10^{-5}$*	2.98 ± 0.07	0.45 ± 0.04
	Optimal Coding	3.59	2.51	$< 10^{-5}$*	2.59	1.
10	LazImpa	10.82 ± 0.28	6.54 ± 0.06	$< 10^{-5}$*	3.87 ± 0.10	0.40 ± 0.005
	Optimal Coding	4.08	2.82	$< 10^{-5}$*	3.08	1.

(a) Variations of vocabulary size `voc_size`. By default, the input distribution is a powerlaw and `max_len` = 30.

`max_len`	System	L_{type}	L_{token}	p_{ZLA}	L_{eff}	ρ_{inf}
30	LazImpa	5.49 ± 0.67	3.78 ± 0.34	$< 10^{-5}$*	2.67 ± 0.07	0.60 ± 0.07
	Optimal Coding	2.96	2.29	$< 10^{-5}$*	1.96	1
20	LazImpa	4.36 ± 0.11	3.12 ± 0.06	$< 10^{-5}$*	2.40 ± 0.08	0.55 ± 0.01
	Optimal Coding	2.96	2.29	$< 10^{-5}$*	1.96	1

(b) Variations of maximum length `max_len`. By default, the input distribution is a powerlaw and `voc_size` = 40.

Distribution	System	L_{type}	L_{token}	p_{ZLA}	L_{eff}	ρ_{inf}
powerlaw	LazImpa	5.49 ± 0.67	3.78 ± 0.34	$< 10^{-5}$*	2.67 ± 0.07	0.60 ± 0.07
	Optimal Coding	2.96	2.29	$< 10^{-5}$*	1.96	1
uniform	LazImpa	4.27 ± 0.37	4.27 ± 0.37	/	2.53 ± 0.09	0.81 ± 0.08
	Optimal Coding	2.96	2.96	/	1.96	1

(c) Variations of input distribution. By default: `voc_size` = 40, `max_len` = 30.

Table 3: Efficiency analysis of LazImpa and Optimal Coding for different set of parameters. L_{type} is the mean message length, L_{token} is the mean weighted message length, p_{ZLA} the ZLA significance score, L_{eff} the effective length and ρ_{inf} the information density. '/' indicates that the metric is not relevant. For p_{ZLA}, '*' indicates that the p-value is significant (< 0.001).

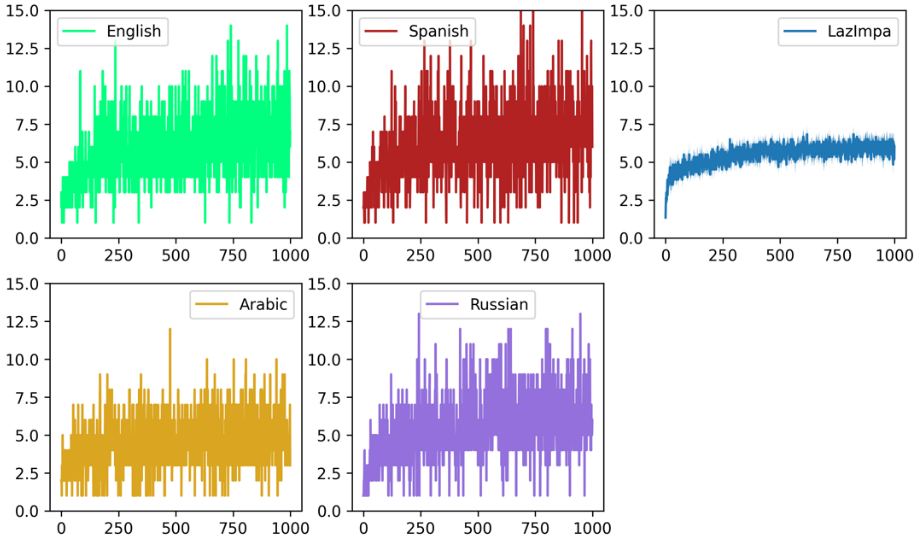


Figure 7: Comparison of the message length as a function of input frequency rank for LazImpa and natural languages.