Catalytic Role Of Noise And Necessity Of Inductive Biases In The Emergence Of Compositional Communication

Łukasz Kuciński∗
Polish Academy of Sciences
lkucinski@impan.pl

Tomasz Korbak
University of Sussex
tomasz.korbak@gmail.com

Pawel Kołodziej
Polish Academy of Sciences†
p.kolodziej@gmail.com

Piotr Miłoś
Polish Academy of Sciences,
University of Oxford,
deepsense.ai
pmilos@impan.pl

Abstract

Communication is compositional if complex signals can be represented as a combination of simpler subparts. In this paper, we theoretically show that inductive biases on both the training framework and the data are needed to develop a compositional communication. Moreover, we prove that compositionality spontaneously arises in the signaling games, where agents communicate over a noisy channel. We experimentally confirm that a range of noise levels, which depends on the model and the data, indeed promotes compositionality. Finally, we provide a comprehensive study of this dependence and report results in terms of recently studied compositionality metrics: topographical similarity, conflict count, and context independence.

1 Introduction

In emergent communication studies, one often considers agents who can share information about a set of objects described by the common features. Such a situation is common in multi-agent systems with partial observation (Foerster et al. (2016), Lazaridou et al. (2017), Jaques et al. (2019), Raczaszek-Leonardi et al. (2018)) and it is the major theme in signaling games (Fudenberg and Tirole (1991), Lewis (1969), Skyrms (2010), Lazaridou et al. (2018)). In a signaling game, one agent (a sender) conveys information about an object to another agent (a receiver), which then has to infer the object’s features. Typically, agents are rewarded if some of the features are correctly identified. During this process, the agents develop a communication protocol. A recent line of work has studied conditions under which compositionality emerges (Batali (1998); Kottur et al. (2017); Choi et al. (2018); Korbak et al. (2019); Li and Bowling (2019); Słowik et al. (2020a,b); Guo et al. (2020)).

Compositionality is a crucial feature of natural languages and it has been investigated extensively in cognitive science (see e.g. Chomsky (1957) Fodor and Pylyshyn (1988)). It is often measured using dedicated metrics such as topographic similarity (Brighton and Kirby (2006); Lazaridou et al. (2018); Kriegeskorte (2008)), context independence (Bogin et al. (2018), conflict count (Kuciński et al. (2020), or positional disentanglement (Chaabouni et al. (2020)). In signaling games it bears a strong resemblance to the concept of disentangled representations, see

∗Corresponding author.
†Now at Google.
In machine learning context, compositionality is perceived as a generalization mechanism (Lake et al. (2017)) and has been used e.g. for goal composition (Jiang et al. (2019)) or knowledge transfer (Li and Bowling (2019)).

In this paper, we theoretically show that inductive biases on both the training framework and the data are needed for compositionality to emerge. A similar observation has been made by Kottur et al. (2017); however, our result is more fundamental and points out a common misconception that compositionality can be learned in a purely unsupervised way. Such a result can be perceived as a discrete analog of Locatello et al. (2019), applicable in the communication context.

We then prove that adding an inductive bias in the loss function coupled with communication over a noisy channel leads to the spontaneous emergence of compositionality. This shows the catalytic role of noise in this process. Intuitively, this can be attributed to the (partial) robustness of compositional language with respect to message corruption caused by a noisy channel.

We experimentally verify that a certain range of noise levels, dependent on the model and the data, promotes compositionality. We provide a wide range of experiments that illustrate the influence of different priors. For the inductive biases in the training framework, we look into the impact of the network architecture as well as implementation and temporal variation in noise. On the data side, we study the effect of scrambling visual input or its description. We also study the generalization properties of the proposed training framework.

2 Related work

The topic of communication is actively studied in multi-agent RL, see Hernandez-Leal et al. (2020, Table 2) for a recent survey. Compositionality is often investigated in the context of signaling games (Fudenberg and Tirole (1991), Lewis (1969), Skyrms (2010), Lazaridou et al. (2018)). Recent research has shown that strong inductive biases or grounding of communication protocols are necessary for the protocol to be compositional (see e.g. Kottur et al. (2017), Słowik et al. (2020b)). The inductive bias can be imposed into the architecture of the agents or the training procedure. For instance, Das et al. (2017) place pressure on agents, to use symbols consistently across varying contexts, by a frequent reset of the agent’s memory. A model-based approach was proposed by Choi et al. (2018) and Bogin et al. (2018), who build upon the obverter algorithm (Oliphant and Batali (1997), Batali (1998)). Słowik et al. (2020a) explore games with hierarchical inputs and shows how agents implemented as graph convolutional networks obtain good generalization. Korbak et al. (2019) implemented the idea of template transfer (Barrett and Skyrms (2017)) by pre-training the agents on simpler subtasks before the target task. Kirby (2001) studied the iterative learning paradigm, where each generation of agents learns the language spoken by the previous generation before starting to communicate. In the machine learning literature, this idea was explored by Li and Bowling (2019), Cogswell et al. (2019) and Ren et al. (2020) with the generation transfer typically implemented as reinitializing the weights of agents’ neural networks. Such an approach inevitably introduces noise into the learning process. This naturally leads to a question of whether the noise itself may be a sufficient mechanism of compositionality, which we will try to address in this paper. Guo et al. (2020) have shown that the choice of a game has a large impact on the properties of a communication protocol emerging in that game, foreshadowing what we call grounding.

The noisy channel model of communication was famously introduced by Shannon (1948). The idea of noise as a driving force in the emergence of communication was first proposed by Nowak and Krakauer (1999), who showed that word-level compositionality is the optimal solution to the problem of communication in a noisy environment under a particular fitness function. Noise is also used in deep learning, e.g. as a regularizer (see e.g. dropout (Srivastava et al., 2014)) or a mechanism allowing backpropagation through a discrete latent (see e.g. Salakhutdinov and Hinton (2009), Kaiser and Bengio (2018)). Noise in the latter context was used in Foerster et al. (2016) in order to learn to communicate. The authors observed that it is essential for successful training.

3 Noisy channel method

We discuss the language and compositionality in Section 3.1. The impossibility result and the need for biases in emergent compositionality is the content of Section 3.2. The communication task considered in this paper as well as the catalytic role of noise is described in Section 3.3. Theoretical results from
We assume that objects with feature space \( F \) contain a combination of these features’ values. For example, the features could represent shape, say squares and circles, and color, say red and green, in which case \( F = \{\text{red square, red circle, green square, green circle}\} \). The features could be identified with partitions of \( F \). More formally, each feature can be defined via equivalence relation, with features values corresponding to equivalence classes of this relation. In our example, the color corresponds to the partition \( \{\text{red square, red circle}\} \) (‘red’) and \( \{\text{green square, green circle}\} \) (‘green’), while for the shape corresponds to the partition \( \{\text{red square, green square}\} \) (‘square’) and \( \{\text{red circle, green circle}\} \) (‘circle’).

We assume that objects with feature space \( F \) can be defined in a space \( X \) and are generated by a two-stage process: first, the feature values are sampled from \( f \in F \), then an element of \( X \) is sampled according to a distribution conditioned on \( f \).

In general, a language is defined as a mapping from objects to strings over some finite alphabet \( A \) (sometimes called messages), \( \ell : X \rightarrow A^* \). In this paper, we will study a subset of languages \( \ell \), where the range of the language has a fixed length, equal to the number of features. We say that \( \ell \) is compositional with respect to a given feature if a change in \( i \)-th feature only impacts a corresponding \( j \)-th index of the message. Continuing the previous example, let \( A = \{a, b\} \) and consider a language \( \ell \) mapping red squares, red circles, green squares, and green circles to \( aa, ba, ab, \) and \( bb \), respectively. Then \( \ell \) is compositional with respect to color and shape features since the change of color only impacts the first index in the message (and analogously for the shape), see Figure 1.a.

Consider a permutation \( \pi : F \rightarrow F \). In our running example, suppose that \( \pi \) is an identity, except that it swaps red circle with green circle, i.e. \( \pi(\text{red circle}) = \text{green circle} \) and \( \pi(\text{green circle}) = \text{red circle} \). The resulting language \( \ell_\pi \) would then map red squares, red circles, green squares, and green circles to \( aa, ba, ab, \) and \( bb \), respectively. This language is not compositional with respect to color and shape features, since if we change a shape value in \( \text{red circle} \), both symbols in the message will change (from \( bb \) to \( aa \)), see Figure 1.b. However, \( \ell_\pi \) is compositional with respect to a different set of features (shape and ‘different color-shape’), see Figure 1.c. Consequently, compositionality should be defined together with features, with respect to which it holds. In the next section, we show that this observation has significant implications for learning.

![Figure 1: Language, features, and compositionality.](image)

### 3.2 Inductive biases and compositionality

In this section, we investigate, whether compositionality can spontaneously emerge in an unsupervised fashion. The answer to this question is negative since the underlying features cannot be inferred from the data.

**Theorem 1.** For a uniform distribution, \( \mu \) on \( F \) and a permutation \( \pi : F \rightarrow F \), the distribution \( \mu \circ \pi^{-1} \) is also uniform.

While this result is elementary to prove (see Appendix F) it has deep implications for the emergence of compositionality in the learning process. Suppose that data represent a balanced set of all features
Any learning process which hopes to achieve compositionality must involve some priors related to the data and learning framework.

Theorem 1 can be viewed as a discrete version of Locatello et al. (2019, Theorem 1). There are multiple ways of grounding the learning process, including imposing inductive biases on the agents and designing loss functions to disentangle features. We study these, together with a new mechanism: injecting noise into the communication channel. In the next section, we show that this mechanism provably achieves compositionality.

### 3.3 Compositionality and communication over a noisy channel

Another major conceptual finding is that compositional communication spontaneously emerges when introducing a relatively simple mechanism – a noisy channel. This is proved in Theorem 2 provided that the loss function penalizes agents’ mistakes, but also rewards for (partially) correct guesses. Recall, that we consider a signaling game, where agents cooperate and develop a communication protocol (a language \( \ell \)) in order to maximize their joint reward. Here a sender observes a certain object with features \( f \in \mathcal{F} \) and sends a message to the receiver to allow him to infer \( f \). The agents are rewarded if some of the features are identified correctly, see equation (1).

The above setup is standard and now we augment it with a noisy channel. The noisy channel is located between the sender and the receiver and may scramble messages. A message \( s \) is transformed into a corrupted message \( s’ \), by replacing each symbol, independently and with probability \( \epsilon \in (0, 1) \), with a different, uniformly sampled, symbol.

For the sake of this section and Theorem 2 below, we will make the following series of assumptions. Let \( \mathcal{F} = \mathcal{F}_1 \times \ldots \times \mathcal{F}_K \), that is the feature space is factorized into \( K \) features. Furthermore, assume that each feature has the same number of values. We will assume that \( \mathcal{X} = \mathcal{F} \) and define a language used by the sender as a mapping \( \ell : \mathcal{F}_1 \times \ldots \times \mathcal{F}_K \rightarrow \mathcal{A}^K \), where \( \mathcal{A} \) is an alphabet and \( |\mathcal{A}| = |\mathcal{F}_i| \). We will further assume that the message \( s = \ell(f) \) is decoded as \( \ell^{-1}(s) \). The corrupted message corresponding to \( f \in \mathcal{F} \) is denoted by \( f’ = \ell^{-1}(\ell(f)) \). The setup for this section and for our experiments (Section 4 and Section 5) differ, see Section 4.1.

Recall that the definition of compositionality (with respect to given features) connects the change in feature values with the change in message symbols. It is thus reasonable to look for loss functions that are somehow factorized in terms of individual symbols. Consider the following loss function:

\[
J(\ell, f) = E[H(\rho(f’, f))],
\]

where \( H \) is a non-negative, strictly increasing function and \( \rho \) is the Hamming distance.

Intuitively, \( \rho \) measures the number of changed features in the corrupted message and \( H \) controls the degree by which we penalize this quantity.

**Theorem 2.** Assume that \( K \geq 2 \), \( \mathcal{F} = \mathcal{F}_1 \times \ldots \times \mathcal{F}_K \), \( |\mathcal{A}| = |\mathcal{F}_i| \geq 2 \), and \( \ell = \mathcal{F} \). Suppose additionally that \( \epsilon < (|\mathcal{A}| - 1)/|\mathcal{A}| \). Then a language \( \ell^* \) minimizes \( J \) over all languages \( \ell \) which are one-to-one mappings if and only if \( \ell^* \) is compositional (with respect to features given by \( \mathcal{F}_i \)).

Informally, Theorem 2 states that optimization of loss function \( J \) promotes compositionality (assumption on the one-to-one property is technical). What makes this possible is the factorized nature of the losses and the introduction of a noisy channel. Interestingly, there exist other loss functions with similar properties. We postpone the analysis of this point and the proof of Theorem 2 to Appendix.

It is instructive to discuss some of Theorem 2 assumptions. Assumption \( \mathcal{F} = \mathcal{X} \) means that \( \ell \) takes semantically meaningful symbolic input. This does not cover many interesting cases, where \( \mathcal{X} \) has representation entangled in terms of features (e.g. an image). Furthermore, the assertion of Theorem 2 holds for a rather wide spectrum of \( \epsilon \) values (up to 0.8 for \(|\mathcal{A}| = 5\), which is what we use in the main experiment in Section 5). This stands in contrast to our experimental results, where we

\( \rho \) is defined as \( \rho(v, w) = \sum_{i=1}^{K} \mathbb{1}(v_i \neq w_i) \).
observe an interaction between different noise levels and compositionality, see Section 5. There is no contradiction here since in Section 5 we study more realistic setup that extends beyond relatively strict assumptions of Theorem 2, see Section 4.1.

4 Experiments setup

4.1 Differences between experimental and theoretical setups

The setup of our experiments is more realistic (and common in this area of research) and extends beyond the assumptions of Theorem 2. We assume that a dataset $X$ contains images, consequently making features entangled in a visual representation (i.e. $F \neq X$). We also allow the alphabet size to differ from the number of feature values, $|A| \neq |F|$. Additionally, the receiver only sees the messages sent by the sender and has to learn to decode the messages. A further gap stems from the implementation details and the use of neural networks, which may converge to suboptimal solutions. We do, however, assume that the feature space $F$ is a Cartesian product (see Section 3.3).

4.2 Training pipeline

In this section, we sketch the training pipeline and postpone the details to Appendix C. The dataset of images observed by the sender is denoted by $D$. Each element of $D$ has $K$ independent features $f_1, \ldots, f_K$ (here we consider $K = 2$). Both the sender and the receiver are modeled as neural networks (for details see Appendix B). The sender network takes an image from $D$ as input and returns a distribution over the space of messages of length $L$ (here we assume $L = K = 2$). We assume that conditionally on the image, the symbols in the message are independent and take values in a finite alphabet $A_s = \{1, \ldots, d_s\}$. This distribution is then distorted by the noisy channel, which is a function that maps probability vectors into $d_s$-dimensional logits. Unless otherwise stated, we assume that noise is be defined as a dense layer with a specific choice of (not learnable) weights matrix and a log activation function:

$$\text{noise}(x) = \log(Wx).$$  (2)

Here we assume that $W \in \mathbb{R}^{d_s \times d_s}$ is a fixed matrix, which takes a probability vector to a probability vector with positive entries. An example of such $W$ is a stochastic matrix. In this paper we use

$$W_{ij} = \begin{cases} 1 - \varepsilon, & i = j, \\ \frac{\varepsilon}{d_s-1}, & i \neq j. \end{cases}$$  (3)

Alternative implementations of noise architecture are possible, see Appendix C. The noisy logits are then used to sample a message and pass it to the receiver. To make this operation differentiable we use Gumbel-Softmax (Jang et al. (2017)) with Straight-Through mode (Kaiser and Bengio (2018)).

Upon receiving the message, the receiver outputs a distribution over possible values of the features, using an alphabet $A_r = \{1, \ldots, d_r\}$. Finally, the neural networks are trained using a linear combination of cross-entropy loss for the receiver, cross-entropy loss for the sender, and $L_2$-regularization. The loss term for the sender incentives the language to be a one-to-one mapping.

4.3 Datasets

We use two datasets: shapes3d (used in Burgess and Kim (2018) and included in the TensorFlow datasets package) and a dataset used by Choi et al. (2018) and Korbak et al. (2019), which we will refer to as the obverter dataset, see Figure 2. The datasets are similar but offer different forms of visual variability. Shapes3d dataset includes images of 3D shapes. Each element is a $(64, 64, 3)$ RGB image, and is characterized by multiple features, such as shape or object hue. We choose images with values for both features ranging in $\{0, 1, 2, 3\}$. The obverter dataset contains images of four shapes (box, cylinder, ellipsoid, sphere) in four colors (blue, cyan, gray, green). Each image has dimensions $(128, 128, 3)$, and we use 1000 images for each shape–color pair. For details see Appendix A.

We believe that Gumbel-Softmax approach is now an established choice in emergent communication research, see e.g. Lee et al. (2018), Mordatch and Abbeel (2018). Chaabouni et al. (2020) report that Gumbel-Softmax converges to similar solutions as REINFORCE but faster and is more stable. The dataset is available at https://github.com/benbogin/obverter. We used the code provided in the repository to generate 1000 images for each color-shape pair.
4.4 Training details

The sender and the receiver are implemented as feed-forward neural networks. To ensure the diversity of random initializations we run each experiment with 100 random seeds. For each seed, there were 100 evaluation runs, once every 2000 network updates. We report metrics averaged over the last 20 evaluation runs. For details on architecture and hyperparameters’ choice, see Appendix B.

4.5 Compositionality measures

We measure compositionality in terms of four metrics used in emergent communication literature: topographic similarity, conflict count, context independence, and positional disentanglement. For all metrics, the higher values the better, except for conflict count, for which the reverse is true. The results presented in Section 5 indicate that the results. Parallel to compositionality metrics, we also report accuracy, which we often refer to as acc.

Topographic similarity

Topographic similarity (Brighton and Kirby, 2006; Lazaridou et al., 2018), or topo for short, is a popular measure of structural similarity between messages and features. Let $L_f : F \times F \rightarrow \mathbb{R}_+$ be a distance over features and $L_{rm} : A_s^m \times A_s^m \rightarrow \mathbb{R}_+$ be a distance over messages. The topographical similarity is the Spearman $\rho$ correlation of $L_f$ and $L_{rm}$ measured over a joint uniform distribution over features and symbols. We choose $L_{rm}$ to be the Levenshtein (1966) distance and treat features $f \in F$ as ordered pairs or features so we can choose $L_f$ to be the Hamming distance. We use topo as the main metric.

Figure 3 shows the expected value of topographic similarity for a random bijective language with a message length equal to 2. For 5-symbol alphabet, this value does not exceed 0.2 which sets a point of reference for compositionality results. For derivation see Appendix E.

Conflict count

Let $\phi : \{1, \ldots, K\} \rightarrow \{1, \ldots, K\}$ be a permutation. The principal meaning of a symbol $s$ at position $j$ is defined as $m(s, j; \phi) = \arg\max_{v} \text{count}(s, j, v; \phi)$, where $\text{count}(s, j, v; \phi)$ is defined as $\sum_{(\text{img}, f) \in D} \mathbb{1}(\text{img})_{j} = s, f_{\phi(j)} = v$, $v$ runs over all values of all features, and ties in $\arg\max$ are broken arbitrarily. Then, conflict count metric, $\text{conf}$ for short, is defined as $\text{conf} = \min_{\phi} \sum_{s,j} \text{score}(s, j; \phi)$, where $\text{score}(s, j; \phi) = \sum_{v \neq m(s, j; \phi)} \text{count}(s, j, v; \phi)$. Intuitively, score measures how many times the feature assigned to a symbol $s$ at a position $j$ diverts from its principal meaning $m(s, j; \phi)$. $\text{conf}$ sums these errors and takes min over possible orderings $\phi$. This metric was introduced in Kuciński et al. (2020).

Context independence

Context independence (Bogin et al. (2018)), abbreviated here as cont, measures the alignment between symbols forming a message and features $f_1, \ldots, f_K$. By $p(s|f)$, we mean the probability that the sender maps a feature $f$ to a message containing symbol $s \in \Sigma$. We define the inverse probability $p(f|s)$ similarly. Finally, we define $s^f = \arg\max_{s} p(f|s)$; $s^f$ is the symbol most often sent in presence of a feature $f$. Then, context independence is $\mathbb{E}(p(s^f|k) \cdot p(k|v^K))$; the expectation is taken with respect to the joint uniform distribution over features and symbols.

Positional disentanglement

Let $s_j$ denote the $j$-th symbol of a message $f(d)$, and $c_i^1$ the feature with the highest mutual information with $s_j$, and $c_i^2$ with the second highest mutual information: $c_i^1 = \arg\max_{c} \mathcal{I}(s_j; c), c_i^2 = \arg\max_{c \neq c_i^1} \mathcal{I}(s_j; c)$ where $\mathcal{I}(\cdot; \cdot)$ is mutual information and $c$ is a feature value. Then, positional disentanglement (Chaabouni et al. 2020) is defined as
Figure 4: Results of the main experiment on shapes3d dataset. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement, and acc for accuracy.

\[ \frac{1}{L} \sum_{j=1}^{L} \left( \mathcal{I}(s_j; c_1) - \mathcal{I}(s_j; c_2) \right) / \mathcal{H}(s_j) \] 

where \( L \) is the maximum message length and \( \mathcal{H}(s_j) \) is entropy over the distribution of symbols at \( j \)-th place in messages for each feature. We ignore positions with zero entropy. We will call this measure \( \text{pos} \), for short.

5 Experiment results

In this section, we study how different inductive biases for the model and the data influence compositionality. The main experiment is presented in Section 5.1 and the experiments in Section 5.2-5.4 are its variation. In the main experiment we assume that the message length is \( K = 2 \), \( |A_s| = 5 \), and \( |A_r| = 8 \) (see Section 4.2). Notice that both \( A_s \) and \( A_r \) differ from one another and from the number of possible feature values (which equals 4). It turns out that the qualitative results are similar for both the shapes3d and obverter datasets, and in the interest of brevity we only report the results for both in the main experiment. Supplementary material for this section can be found in Appendix D.

Our main findings include the fact that the noise indeed catalyzes the emergence of compositionality. There is an interesting dependence on the noise level: too high noise may impede learning, while too small noise vanishes within other sources of noise. Importantly, this phenomenon appears consistently across different biases. Using two head output of the network is the strongest bias toward compositionality (amongst studied). Finally, we show that compositionality can generalize to unseen cases when fine-tuning is allowed.

5.1 Main experiment: emergence of compositionality

The results, presented in Figure 4, illustrate how noise catalyzes the emergence of compositionality. The top panel of Figure 4 shows the important patterns for metrics: they improve until an extremal point is reached, and decline afterward. Topographic similarity achieves extremum for noise level 0.1, reaching value 0.87, which is a significant improvement upon 0.79 for the lack of noise. The accuracy drops down with an increase of the noise level, as expected, however the speed of the decline increases. This shows that there is an interesting compositionality-accuracy trade-off. The bottom panel of Figure 4 complements the overall picture with a visualization of metrics’ distribution. We see interesting dynamics in topographic similarity distribution with respect to change in the noise levels. Namely, it starts by accumulating mass at the higher spectrum of its values, reaching a peak for noise 0.1, after which it transitions to a bimodal distribution, finally shifting its mass more towards the mediocre end of the spectrum. An interesting observation is that the results for experiments with achieved high accuracy, are not only better but also the effect of noise is more pronounced (for

---

6Positional disentanglement is related to residual entropy proposed by Resnick et al. (2020). Chaabouni et al. (2020) also proposed bag-of-words disentanglement, which assumes order-invariance of messages. Due to our architecture choice, this assumption is not met, hence we decided not to report this metric.
Figure 5: Topographic similarity for experiments in Sections 5.2 and Section 5.3. The shaded areas correspond to bootstrapped 95%-confidence intervals for average topo.

instance there are 47 seeds with accuracy exceeding 0.9, for which noise 0.16 yields 0.96 topo). For the sake of brevity, we defer the discussion of this phenomenon to Appendix D.1. The accuracy undergoes a similar transformation as a topographic similarity. The detailed numerical analysis, as well as corresponding results for the obverter dataset, can be found in Appendix D.1.

5.2 Influence of model inductive bias

Different number of symbols Here we study the impact of a different number of symbols on compositionality. In our experiments we used the communication channel with \(|A_s| = 5\), giving a total of \(25 = (5 \times 5)\) possible messages. This is slightly redundant since only \(16 = (4 \times 4)\) is required, so this is the first case that we study here. It turns out, that allowing for only 16 messages makes the training less stable. For topographic similarity, see Figure 5(a), small to medium values of noise exhibit wide confidence intervals and it is statistically hard to distinguish between the metric values (this might be attributed to a bimodal distribution of topo in this noise range, see Figure 15 in Appendix D.2). For larger values of noise (greater than 0.16), topo starts to visibly decline. As the second experiment, we considered the total of \(64 = (8 \times 8)\) messages. Interestingly, the topographic similarity values for the small noise regime (up to 0.08) do not improve over the baseline value (0.79), see Figure 5(b). This behavior changes for medium to large values of noise, where we can observe a visible increase in topo, peaking at 0.88 with a noise level of 0.24. For details see Appendix D.2.

Variable noise Understanding what happens under varying noise is an interesting and subtle problem. It would most probably arise in more complex situations when a communication channel is a part of a bigger system. Additionally, it might be similar to a phenomenon observed in supervised learning, indicating that training can benefit from learning rate warmup (see e.g. Goyal et al. (2017), Frankle et al. (2020)). In this paragraph, we discuss a simple experiment with increasing noise and leave a more nuanced study for further work. More precisely, in the initial stage of training the noise is kept at the level \(\epsilon_0\), and after \(T \in \{200, 700, 1200, 1700, 2200, 2700\}\) network updates it is switched to a different value, \(\epsilon_T\), and kept there for the rest of the training. The results for topographic similarity are presented in Figures 5(c)-(d), where \(\epsilon_0 = 0\) and \(\epsilon_T = 0.1\) and \(\epsilon_T = 0.15\), respectively. In Figure 5(d) we see that topo increases from 0.85 for \(T = 200\), to 0.9 for \(T = 2700\). The effect in Figure 5(c) is weaker and the variance in the results is quite high. The details can be found in Appendix D.3.

Sensitivity with respect to small architecture changes This set of experiments aims to check the impact of changing the parameters of CNN and the dense layers in the agents’ network. In the former experiment, we change the number of filters in the sender’s CNN architecture from two layers with 8 filters, to two layers with 16 filters. This results in the slight change of topographic similarity
Fine-tuning for a baseline agent trained with noise 0.1 and with several features removed from the training set (the diagonal from the shape-by-color matrix). On the x-axis is the number of finetuning updates. See Figure 5(e), with the highest average value of 0.86 for noise range 0.14 significantly outperforming the zero-noise case (0.79). For the second experiment, we added a dense layer (with 64 neurons) to the receiver’s architecture, see Figure 5(f). This again shows improvement of compositionality due to noise, although the noise in the range [0.08, 0.16] performs roughly the same (see also Appendix D.4).

**Sensitivity to noisy channel implementation** Here we consider an alternative noisy channel implementation, where the noise permutes the sampled symbols, as opposed to distorting the distribution of the symbols (see Appendix C). It turns out that with this change, the scale of noise where the interesting things happen changes as well, see Figure 5(g). Namely, when compared with the main experiment, noise values are in the range smaller by the order of magnitude (we report results for noise levels in \{0.000, 0.001, ... , 0.015\}. The results suggest that the small values of noise can help, while the larger noise levels lead to a decline in compositionality, see also Appendix D.5.

**Longer message** In this section, we discuss the experiment with an additional feature (with floor color acting as the third feature) and a message of length 3. The result for topographic similarity can be seen in Figure 5(h), see also Appendix D.6. The overall topographic similarity level is lower than in the main experiment, however, the distinctive peak is visible, here for noise level 0.08.

### Data inductive biases

**Visual priors** This experiment was intended to check how much the CNN-backed input is relevant in the compositionality context. We could conjecture that CNN may facilitate shape recognition and therefore be the driving force in the emergence of languages compositional with respect to the canonical shape, color split. To check this we impair the prior by scrambling images, as depicted in the bottom panel Figure 2. An image is scrambled by splitting it into \((64/x)\times2\) disjoint tiles of height and width equal to \(x\), and randomly reshuffling them. This procedure significantly distorts the accuracy profile of the method (see Appendix D.7). In particular, each transition from coarser to finer tiles, the accuracy decreases significantly: for zero-noise it drops from 0.97 for no tiling, to 0.95 for \(x = 32\), to 0.79 for \(x = 16\) and 0.53 for \(x = 8\). The overall compositionality metrics decrease as well, but the characteristic peak for some positive noise level is still present, see Figure 5(i)-(k). Having said that, the metrics incur a significant boost, when computed for a subset of experiments with high accuracy. We conjecture that the explanation is that the CNN prior is indeed relevant but is not the only one (the output considered in the next section is another).

**Scrambled labels** In this experiment, we aimed to understand how much the overall architecture output is important in the emergence of compositionality. The receiver’s network has a two-headed output and the training framework uses a factorized loss function. In the standard setting, we compare the heads’ outputs with ‘color’ and ‘shape’ respectively, therefore we reflect human priors from the data. In this experiment, we distort this setting, permuting the set of (color, shape) and factorizing them into new labels (see Appendix D.8). We use a random permutation, so the new labels are abstract and correspond to some joint color-shape concepts. In our experiments, we show that languages, which emerge are compositional with respect to these new concepts, see Figure 5(l). Consequently, they are not compositional in the standard color-shape framework. This is in line with the claims of Section 3.2 and further highlights that the output inductive bias is essential.

### Generalization

**Network features** In this paragraph, we present visualizations of the sender network, to gain some insights into what the network is doing, and whether we can observe any obvious overfit. In the
upper panel of Figure 7, a t-SNE (Van der Maaten and Hinton (2008)) visualization is shown for both the raw shapes3d dataset and the last dense layer of a trained agent with noise 0.1. We observe that the network successfully disentangles features of the data, which in the raw dataset appears to be entangled. This could be a good sign in the context of generalization.

Furthermore, Figure 7 (bottom panel) shows occlusion and saliency maps for a sample image from shapes3d dataset. The network seems to pay attention to parts of the image that could be relevant for the task.

**Zero-shot and fine-tuning** In this experiment, we study the generalization properties of the training protocol. We trained the baseline agent with noise 0.1 with several features removed from the training set (a diagonal from the shape-by-color matrix) and used the removed set to analyze the out-of-distribution performance. In the zero-shot setup, the messages on these observations were not in line with the compositional structure acquired on the training set, which is reflected by rather unimpressive outcomes for each metric (initial values for each metric presented in Figure 6). This may suggest that for a zero-shot generalization stronger biases might be required (see e.g. Słowik et al. (2020a,b)). However, after a few network updates on the whole dataset (without the removed diagonal) we observed a quite significant increase in compositionality metrics, see Figure 6. The highest improvement can be seen roughly in the first 50 additional network updates.

**6 Limitations of the method**

Here we summarize a few limitations of the method, that we hope to overcome in future research.

**Theory** Theorem 2 uses several restrictive assumptions, e.g. fixed message length or alphabet size equal to the range of feature values. In a general setting, it could be the case that noise would promote non-compositional, error-correcting, communication protocols. Stating the general conditions for the emergence of compositionality is an open problem.

**Choice of datasets** Confirmation on non-synthetic datasets is needed to fully underpin our method.

**Communication protocol** We assume a simple communication protocol with messages of length two and two features, each taking four values (except for experiments in Section 5.2).

**Compositionality model** We assume a rather simple compositionality model based of independent features. Exploring non-trivial compositionality (Steinert-Threlkeld, 2020; Korbak et al., 2020) might be an important conceptual development.

**Architecture** Our architecture might implicitly exhibit some unknown biases that increase compositional. On the other hand, the architecture might be too simple to achieve strong results in tasks such as zero- or few-shot generalization.

**Multi-agent interactions** We used supervised training setup, it would be natural to test the influence of noise in reinforcement learning scenarios.

**7 Conclusions**

In this paper, we theoretically show that inductive biases on both the training framework and the data are needed for the compositionality to emerge spontaneously in signaling games. We then formulate inductive biases in the loss function and prove that they are sufficient to achieve compositionality when coupled with communication over a noisy channel. Consequently, we highlight the catalytic role of noise in the emergence of compositionality. We perform a series of experiments in order to understand different aspects of the proposed framework better. We empirically validate that, indeed, a certain range of noise levels, dependent on the model and the data, promotes compositionality. Our work is foundational research and does not lead to any direct negative applications.

---

6 It is known that t-SNE does not necessarily represent cluster sizes, distances, and respective positions (see Wattenberg et al., 2016). Hence, t-SNE cannot be expected to serve as a compositionality metric.
Acknowledgments and Disclosure of Funding

The work of Piotr Miłoś was supported by the Polish National Science Center grant UMO-2017/26/E/ST6/00622. The work of Tomasz Korbak was supported by the Leverhulme Doctoral Scholarship. We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Centers: ACK Cyfronet AGH, PCSS) for providing computer facilities and support within computational grant no. PLG/2019/012498. Our experiments were managed using https://neptune.ai. We would like to thank the Neptune team for providing us access to the team version and technical support.

References

Barrett, J. A. and Skyrms, B. (2017). Self-assembling Games. The British Journal for the Philosophy of Science, 68(2):329–353.

Batali, J. (1998). Computational simulations of the emergence of grammar. Approach to the Evolution of Language, pages 405–426.

Bogin, B., Geva, M., and Berant, J. (2018). Emergence of Communication in an Interactive World with Consistent Speakers. arXiv preprint arXiv:1809.00549.

Bouchacourt, D. and Baroni, M. (2018). How agents see things: On visual representations in an emergent language game. Empirical Methods in Natural Language Processing.

Brighton, H. and Kirby, S. (2006). Understanding Linguistic Evolution by Visualizing the Emergence of Topographic Mappings. Artificial Life, 12(2):229–242.

Burgess, C. and Kim, H. (2018). 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/.

Chaabouni, R., Kharitonov, E., Bouchacourt, D., Dupoux, E., and Baroni, M. (2020). Compositional-ity and Generalization In Emergent Languages. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4427–4442.

Choi, E., Lazaridou, A., and de Freitas, N. (2018). Compositional Obverter Communication Learning From Raw Visual Input. In International Conference on Learning Representations.

Chomsky, N. (1957). Syntactic structures. Mouton de Gruyter.

Cogswell, M., Lu, J., Lee, S., Parikh, D., and Batra, D. (2019). Emergence of Compositional Language with Deep Generational Transmission. arXiv preprint arXiv:1904.09067.

Das, A., Kottur, S., Moura, J. M., Lee, S., and Batra, D. (2017). Learning cooperative visual dialog agents with deep reinforcement learning. In Proceedings of the IEEE international conference on computer vision, pages 2951–2960.

Fodor, J. A. and Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: a critical analysis. Cognition, 28(1-2):3–71.

Foerster, J. N., Assael, Y. M., de Freitas, N., and Whiteson, S. (2016). Learning to Communicate with Deep Multi-Agent Reinforcement Learning. NIPS’16 Proceedings of the 30th International Conference on Neural Information Processing Systems.

Frankle, J., Dziugaite, G. K., Roy, D., and Carbin, M. (2020). Linear mode connectivity and the lottery ticket hypothesis. In International Conference on Machine Learning, pages 3259–3269.

Fudenberg, D. and Tirole, J. (1991). Game Theory. MIT Press.

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrrola, A., Tulloch, A., Jia, Y., and He, K. (2017). Accurate, large minibatch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:1706.02677.

Guo, S., Ren, Y., Słowiński, A., and Mathewson, K. (2020). Inductive bias and language expressivity in emergent communication. arXiv preprint arXiv:2012.02875.
Hernandez-Leal, P., Kartal, B., and Taylor, M. E. (2020). A very condensed survey and critique of multiagent deep reinforcement learning. In Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, pages 2146–2148.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Giori, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2017). beta-VAE: Learning basic visual concepts with a constrained variational framework. In International Conference on Learning Representations.

Jang, E., Gu, S., and Poole, B. (2017). Categorical reparametrization with gumbel-softmax. In Proceedings International Conference on Learning Representations 2017.

Jaques, N., Lazaridou, A., Hughes, E., Gulcehre, C., Ortega, P., Strouse, D., Leibo, J. Z., and De Freitas, N. (2019). Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning. In International Conference on Machine Learning, pages 3040–3049.

Jiang, Y., Gu, S., Murphy, K., and Finn, C. (2019). Language as an abstraction for hierarchical deep reinforcement learning. In Advances in Neural Information Processing Systems, volume 32.

Kaiser, L. and Bengio, S. (2018). Discrete autoencoders for sequence models. arXiv preprint arXiv:1801.09797.

Kim, H. and Mnih, A. (2018). Disentangling by factorising. In International Conference on Machine Learning, pages 2649–2658.

Kirby, S. (2001). Spontaneous evolution of linguistic structure-an iterated learning model of the emergence of regularity and irregularity. IEEE Transactions on Evolutionary Computation, 5(2):102–110.

Korbak, T., Zubek, J., Kuciński, Ł., Miłoš, P., and Raczaszek-Leonardi, J. (2019). Developmentally motivated emergence of compositional communication via template transfer. arXiv preprint arXiv:1910.06079.

Korbak, T., Zubek, J., and Raczaszek-Leonardi, J. (2020). Measuring non-trivial compositionality in emergent communication. arXiv preprint arXiv:2010.15058.

Kottur, S., Moura, J. M., Lee, S., and Batra, D. (2017). Natural Language Does Not Emerge ‘Naturally’ in Multi-Agent Dialog. arXiv preprint arXiv:1706.08502.

Kriegeskorte, N. (2008). Representational similarity analysis – connecting the branches of systems neuroscience. Frontiers in Systems Neuroscience.

Kuciński, Ł., Kołodziej, P., and Miłoš, P. (2020). Emergence of compositional language in communication through noisy channel. ICML 2020 Workshop LaReL.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. (2017). Building Machines That Learn and Think Like People. Behavioral and brain sciences, 40.

Lazaridou, A., Hermann, K. M., Tuyls, K., and Clark, S. (2018). Emergence of linguistic communication from referential games with symbolic and pixel input. In International Conference on Learning Representations.

Lazaridou, A., Peysakhovich, A., and Baroni, M. (2017). Multi-Agent Cooperation and the Emergence of (Natural) Language. In International Conference on Learning Representations.

Lee, J., Cho, K., Weston, J., and Kiela, D. (2018). Emergent translation in multi-agent communication. In International Conference on Learning Representations.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In Soviet physics doklady, volume 10, pages 707–710. Soviet Union.

Lewis, D. K. (1969). Convention: a philosophical study. Blackwell.

Li, F. and Bowling, M. (2019). Ease-of-teaching and language structure from emergent communication. In Advances in Neural Information Processing Systems, pages 15825–15835.
Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B., and Bachem, O. (2019). Challenging common assumptions in the unsupervised learning of disentangled representations. In International Conference on Machine Learning, pages 4114–4124.

Mordatch, I. and Abbeel, P. (2018). Emergence of grounded compositional language in multi-agent populations. In Thirty-second AAAI conference on artificial intelligence.

Nowak, M. A. and Krakauer, D. C. (1999). The evolution of language. Proceedings of the National Academy of Sciences, 96(14):8028–8033.

Oliphant, M. and Batali, J. (1997). Learning and the Emergence of Coordinated Communication. Center for Research on Language Newsletter, 11.

Raczaszek-Leonardi, J., Nomikou, I., Rohlfing, K. J., and Deacon, T. W. (2018). Language Development From an Ecological Perspective: Ecologically Valid Ways to Abstract Symbols. Ecological Psychology, 30(1):39–73.

Ren, Y., Guo, S., Labeau, M., Cohen, S. B., and Kirby, S. (2020). Compositional languages emerge in a neural iterated learning model. In International Conference on Learning Representations.

Resnick, C., Gupta, A., Foerster, J., Dai, A. M., and Cho, K. (2020). Capacity, Bandwidth, and Compositionality in Emergent Language Learning. In Autonomous Agents and Multiagent Systems, pages 1125–1133.

Salakhutdinov, R. and Hinton, G. (2009). Semantic hashing. International Journal of Approximate Reasoning, 50(7):969–978.

Shannon, C. E. (1948). A mathematical theory of communication. The Bell system technical journal, 27(3):379–423.

Skyrms, B. (2010). Signals: evolution, learning, & information. Oxford University Press, Oxford ; New York.

Slowik, A., Gupta, A., Hamilton, W. L., Jamnik, M., and Holden, S. B. (2020a). Towards graph representation learning in emergent communication. arXiv preprint arXiv:2001.09063.

Slowik, A., Gupta, A., Hamilton, W. L., Jamnik, M., Holden, S. B., and Pal, C. (2020b). Exploring structural inductive biases in emergent communication. arXiv preprint arXiv:2002.01335.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1):1929–1958.

Steinert-Threlkeld, S. (2020). Towards the Emergence of Non-trivial Compositionality. Philosophy of Science.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. Journal of machine learning research, 9(11).

Wattenberg, M., Viséas, F., and Johnson, I. (2016). How to use t-sne effectively. Distill.
A Datasets

A.1 Shapes3d

Shapes3d is a dataset (see Burgess and Kim (2018) and the Tensorflow Datasets package) consisting of $64 \times 64 \times 3$ RGB images of objects having six independent features (floor color, wall color, object color, scale, shape, and orientation), see Figure 2. In this paper, we use four shapes (cube, cylinder, sphere, and rounded cylinder) and four object colors (red, orange, yellow, and green), totaling 192000 images.

A.2 Obverter

The obverter dataset (Bogin et al. (2018)) is available at the following address: https://github.com/benbogin/obverter. The original dataset consists of $128 \times 128 \times 3$ RGB images of objects having eight colors and five shapes. In this paper, we used four colors (blue, cyan, gray, green) and four shapes (box, cylinder, ellipsoid, sphere), see Figure 2. We have generated 1000 samples for each color-shape combination using a generation script available at the dataset repository, hence the total number of images is 160000. Since the qualitative results were similar, and in the interest of brevity, we report results for the obverter only for the main experiment (Appendix D.1).

B Experimental setup

B.1 Architecture

The network consists of three main parts: the sender, the receiver, and the noisy discrete channel between them see Figure 8. The sender network consists of two convolutional layers (with 8 filters, kernel 3 × 3, stride 1, and elu activation function), each coupled with a 2 × 2 max pool layer with stride 2. The last max pool layer’s output is passed through two dense layers (with 64 neurons and elu activation) and a linear classifier with softmax for each symbol. The noisy channel layer consists of a dense layer with $|A_s|$ neurons, a fixed weights matrix, and a log activation function. This is followed by a Gumbel softmax layer. The receiver network takes two encoded symbols as input and concatenates them to obtain one input vector $s$. Consequently, $s$ and $1 - s$ are passed to the dense layers, the result is summed up and processed by the elu activation function and two dense layers (similarly to Kaiser and Bengio (2018)). There are two linear classifiers with a softmax layer at the output: one for the shape and one for the color. Each dense layer in the receiver has 64 neurons.

B.2 Hyperparameters and training

For training, we used $\lambda_{KL} = 0.01$, $\lambda_s = 0.0003$, an Adam optimizer (with $\beta_1 = 0.9$, $\beta_2 = 0.999$), learning rate 0.0001, and a batch size of 64. The same set of hyperparameters was used for all the experiments. The hyperparameters were chosen on the original obverter dataset available at the repository referenced in Appendix A.2. We used a grid search over parameters: learning rate ($1e - 2, 1e - 3, 1e - 4, 3e - 4$), kl regularization coefficient ($1e - 1, 1e - 2, 2e - 2, 3e - 2, 1e - 3, 3e - 3, 5e - 3, 1e - 4$), the number of CNN’s filters (8, 16), the CNN’s filter sizes ($3 \times 3, 5 \times 5$), the sender’s embedding size (32, 64), $l_2$ regularizer weight ($1e - 2, 1e - 3, 3e - 3, 1e - 4, 3e - 4, 1e - 6$), and the number of neurons in receiver’s dense layers (32, 64).

Each experiment was run on 100 seeds and had 200000 network updates. The dataset was split into the training set (90% of the total) and on the test set (remaining 10% of the dataset). The evaluation was done every 2000 updates on the test set.

B.3 Infrastructure used

The typical configuration of a computational node used in our experiments was: the Intel Xeon E5-2697 2.60GHz processor with 128GB memory. On a single node, we ran 4 or 5 experiments at the same time. A single experiment (single seed) takes about 5 hours. We did not use GPUs; we found that with the relatively small size of the network (see Appendix B.1), it offers only slight wall-time improvement while generating substantial additional costs.
C Training pipeline

C.1 Detailed derivation

The dataset of images observed by the sender is denoted by $\mathcal{D}$. Each element of $\mathcal{D}$ has $K$ independent features $f_1, \ldots, f_K$ (here we consider $K = 2$). Both the sender and the receiver are modeled as neural networks ($s_\theta$ and $r_\psi$, respectively; for details see Appendix B). The sender network takes an image from $\mathcal{D}$ as input and returns a distribution over the space of messages of length $L$ (here we assume $L = K = 2$). We assume that conditionally on the image, the symbols in the message are independent and take values in a finite alphabet $A_s = \{1, \ldots, d_s\}$. We furthermore assume that features are enumerated with $A_f = \{1, \ldots, d_f\}$ and the receiver’s alphabet is $A_r = \{1, \ldots, d_r\}$.

Formally, $s_\theta(img) = (s_{i,\theta}(img))_{i=1}^K$, where $s_{i,\theta}(img) = (s_{i,j,\theta}(img))_{j=1}^{d_s} \in \mathcal{P}(A_s)$ represents the probability distribution corresponding to the $i$th symbol. Define a function $\text{noise}: \mathcal{P}(A_s) \to \mathbb{R}^{d_s}$ as follows:

$$\text{noise}(x) = \log(Wx),$$

(4)

where $W \in \mathbb{R}^{d_s \times d_r}$ is a fixed matrix, such that $Wx > 0$ and $Wx \in \mathcal{P}(A_s)$, for any $x \in \mathcal{P}(A_s)^d$.

The second condition on $W$ is satisfied, for instance, by a family of stochastic matrices; several examples are also given at the end of this section. In this paper, we use $W$ defined as

$$W_{ij} = \begin{cases} 1 - \varepsilon, & i = j, \\ \frac{\varepsilon}{d_s - 1}, & i \neq j. \end{cases}$$

Let $\tilde{s}_\theta(img)$ denote the logits of the symbol distribution which passes through the noisy channel:

$$\tilde{s}_\theta(img) = \text{noise}(s_\theta(img)).$$

Suppose further that $g^i = (g_{i1}, \ldots, g_{id_s})$ is a vector of i.i.d. Gumbel(0, 1) random variables and define the following functions:

$$\text{gumbel\_sample}(x; g) = \arg\max_i (x_i + g_i),$$

$$\text{gumbel\_softmax}(x; \tau, g)i = \frac{\exp((x_i + g_i)/\tau)}{\sum_{j=1}^K \exp((x_j + g_j)/\tau)}.$$

We could also define $\text{noise}$ for all $x \in \mathbb{R}^m$, for some $m$, by first applying $\text{softmax}$ to $x$, and then using equation (4).

---

Footnotes:

8$\mathcal{P}(A) = \{p \in \mathbb{R}^{|A|} : p_i \geq 0, \sum_{i \in A} p_i = 1\}$.

9We could also define $\text{noise}$ for all $x \in \mathbb{R}^m$, for some $m$, by first applying $\text{softmax}$ to $x$, and then using equation (4).
Let
\[ \hat{m}_i = \text{gumbel\_softmax}(s_i^i(img); \tau, g^i) \in \mathbb{R}^{d_s}. \]

The receiver neural network is denoted as
\[ r_{\psi}(m) = (r_{\psi}^i(m))_{i=1}^K, \]
where \( r_{\psi}^i(m) = (r_{\psi}^i_j(m))_{j=1}^{d_s} \in \mathcal{P}(A_s) \) represents the probability distribution on \( A_s \), corresponding to the \( i \)th feature.

In the Straight-Through mode (see Jang et al. (2017)), \( r_{\psi} \) takes \( b_m \) as input half of the time, and the remaining half of the time, it takes \( e_m \). Here
\[ e_m = \text{stop\_gradient}(m - \hat{m}) + \hat{m}, \]
\[ m_i = \text{one\_hot}(\omega_i) \in \mathbb{R}^{d_s}, \]
\[ \omega_i = \text{gumbel\_sample}(s_i^i(img); g^i) \in A_s, \]
i.e. \( (\omega_1, \ldots, \omega_K) \) is a sampled noisy message. The neural networks are trained using the following loss function:
\[ L = L_{xent} + \lambda_{KL} L_{KL} + \lambda_{l_2} L_{l_2}. \]

The cross-entropy loss is defined as
\[ L_{xent} = -\mathbb{E}_{(img, f_1, \ldots, f_K) \sim \mathcal{D}} \left[ \sum_{i=1}^K \log r_{f_i, \psi}(\hat{m}(img)) \right]. \]

Furthermore, \( L_{KL} = \mathbb{E}_{x \sim \mathcal{D}} \left[ \sum_{i=1}^K \text{KL}(U(A_s) || s_i^0(x)) \right] \) and \( L_{l_2} = ||\theta||_2 + ||\psi||_2 \), where \( U(A_s) \) denotes the uniform distribution of \( A_s \). The KL loss incentivizes the language to be a one-to-one mapping.

C.2 Alternative noise architecture

The above implementation of noise is not the only one possible. We can apply noise after the message is formed. Denote the uncorrupted sender’s message
\[ m_i = \text{gumbel\_softmax}(s_i^0(img); \tau, g^i) \in \mathbb{R}^{d_s}. \]

Further, let \( \rho_i \) be uniformly sampled random permutations of \( A_s \) and \( \epsilon_i \) are Bernoulli random variables such that \( \mathbb{P}(\epsilon_i = 1) = \epsilon = 1 - \mathbb{P}(\epsilon_i = 0) \), with \( \epsilon > 0 \). We define the noise matrices
\[ N_i = \epsilon_i P^{\rho_i} + (1 - \epsilon_i) I, \]

where \( P^{\rho_i} \) is the permutation matrix corresponding to \( \rho_i \) and \( I \) is the identity matrix. The corrupted message is then given by
\[ \hat{m}_i = N_i m_i. \]

Above we assume that \( \hat{m}_i, m_i \) is encoded in the one-hot vector form \( \in \mathbb{R}^{d_s} \). We note that this particular implementation of the noise has the advantage of being differentiable using the standard autograd methods.

D Detailed results

Each experiment was run on 100 seeds. When presenting results we give 95%-confidence intervals, bootstrapped using 4000 resamples.

D.1 Main experiment

The results for the shapes3d dataset are visualized in Figure 9 (which is the copy of Figure 4 placed here for convenience), and numerical results are summarized in Table 1. Similarly, for the obverter dataset, the results can be found in Figure 10 and in Table 2. From the qualitative perspective, the results for the obverter dataset are similar to the ones obtained for the shapes3d dataset. Having said

\[ \text{\footnotesize We cannot differentiate the random sampling of } \rho_i, \epsilon_i \text{ but we can differentiate multiplication with respect to } N_i. \]

10
that, we can see that overall the metrics are more stable (resulting in narrower confidence intervals),
the best performing noise level is slightly different (0.12), and the density evolution of topo across
different noise levels appears to be smoother.

Figure 9: Shapes3d dataset. Top panel: average value of metrics for various noise levels. The shaded area
 corresponds to bootstrapped 95%-confidence intervals. Bottom panel: kernel density estimators for metrics and
noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context
independence, pos for positional disentanglement and acc for accuracy.

Figure 10: Obverter dataset. Top panel: average value of metrics for various noise levels. The shaded area
 corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators
for metrics and noise levels across seeds. Bottom panel: kernel density estimators for metrics and noise levels
across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence,
pos for positional disentanglement and acc for accuracy.

As observed in Section 5.1, there is an interesting and non-trivial interplay between compositionality
and accuracy. In Figure 11 we visualize topographic similarity behavior when conditioned on
experiments with high accuracy (for shapes3d; for the obverter see Figure 12). We can see an increase
in the metrics values across all noise levels with the increase of accuracy. For example, for noise 0.1 and
threshold 0.85, topo equals 0.91, four percentage points higher than for unconditional case. We
can see that increasing the threshold also strengthen the impact of noise on compositionality (which
can be seen by increasing the profile of topo plots). Additionally, the count curves for smaller noises
dominate the ones for higher noises. Notice, however, that the number of experiments exceeding
some accuracy threshold declines as the threshold increases. For example, there are 80 experiments
with noise level 0.1 exceeding the threshold of 0.85 accuracy, but only 4 with noise level 0.3. This
implies that conclusions for high accuracy thresholds should be treated with care.
Figure 11: Shapes3d dataset. Top left: The values of topo computed for each noise level (hues) and on seeds exceeding a certain accuracy threshold (x-axis). Vertical dashed lines represent three cross-sections, visualized in the bottom panel. Top right: Similar to the left panel, but instead of topo we visualize the number of seeds with accuracy at least as a given threshold (x-axis). Vertical dashed lines represent three cross-sections, visualized in the bottom panel. Bottom: Each of the plots represents a cross-section of the plots in the top panel, taken at points 0.80, 0.85, and 0.90, respectively. On the left axis of each figure is the range of topo, whereas on the right axis is the number of seeds with accuracy exceeding the corresponding level. On the x-axis are the noise levels. The scatter plot with 95%-confidence intervals represents the values of topo. The gray dashed line represents the number of seeds with accuracy exceeding a given threshold, for each of the noise levels.

Figure 12: Obverter dataset. Top left: The values of topo computed for each noise level (hues) and on seeds exceeding a certain accuracy threshold (x-axis). Vertical dashed lines represent three cross-sections, visualized in the bottom panel. Top right: Similar to the left panel, but instead of topo we visualize the number of seeds with accuracy at least as a given threshold (x-axis). Vertical dashed lines represent three cross-sections, visualized in the bottom panel. Bottom: Each of the plots represents a cross-section of the plots in the top panel, taken at points 0.80, 0.85, and 0.90, respectively. On the left axis of each figure is the range of topo, whereas on the right axis is the number of seeds with accuracy exceed the corresponding level. On the x-axis are the noise levels. The scatter plot with 95%-confidence intervals represents the values of topo. The gray dashed line represents the number of seeds with accuracy exceeding a given threshold, for each of the noise levels.

For the main experiment, we also provide pair-plots for all metrics and three noise levels: 0.0, 0.1, and 0.2, see Figure 13 and Figure 14 for the shapes3d and the obverter datasets, respectively. It shows the Spearman correlation between metrics, broken down to noise levels (color-coded circles in the
upper triangle of the grid) as well as for the entire group (white circle in the upper triangle of the grid). We can see in Figure 13 that the metrics are highly correlated (the negative correlation with conflict count follows by definition of the metric, see Section 4.5). For zero-noise (blue color), we see that accuracy is high irrespective of the compositionality metrics, resulting in an almost vertical line. Looking at the topo-acc cell, we can see that for mediocre accuracy values, the noise level 0.2 (yellow color) tends to score higher in topo metrics than the noise level 0.1 (green color). This relation reverses for accuracy values closer to 1.0. For topo-conf and topo-pos cells, we see a visible linear correlation, with the effect weakening slightly for higher noise levels. At the topo-cont cell, a lot of the mass of all noise levels is occupied in the center, but the noise level 0.1 more frequently stays in the upper right corner. Similar observations can be done for Figure 14.

Figure 13: Shapes3d dataset. The lower triangle of the grid: color-coded scatter plot for experiments with different noise levels. The upper triangle of the grid: visualization of Spearman correlation between metrics. The large circle with white fill shows the correlation of metrics value without a split into noise levels. The smaller color-coded circles represent the in-group correlation. Diagonal: kernel density estimators for each metric and noise level. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
Figure 14: Obverter dataset. The lower triangle of the grid: color-coded scatter plot for experiments with different noise levels. The upper triangle of the grid: visualization of Spearman correlation between metrics. The large circle with white fill shows the correlation of metrics value without a split into noise levels. The smaller color-coded circles represent the in-group correlation. Diagonal: kernel density estimators for each metric and noise level. Here *topo* stands for topographic similarity, *conf* for conflict count, *cont* for context independence, *pos* for positional disentanglement and *acc* for accuracy.
| noise | topo  | conf  | cont  | pos   | acc  |
|-------|-------|-------|-------|-------|------|
| 0.00  | 0.79  | 4.08  | 0.168 | 0.57  | 0.97 |
|   | [0.76, 0.82] | [3.21, 4.94] | [0.161, 0.175] | [0.51, 0.63] | [0.96, 0.98] |
| 0.02  | 0.80  | 4.26  | 0.166 | 0.52  | 0.98 |
|   | [0.78, 0.82] | [3.46, 5.08] | [0.160, 0.172] | [0.47, 0.58] | [0.98, 0.98] |
| 0.04  | 0.80  | 4.78  | 0.165 | 0.51  | 0.95 |
|   | [0.77, 0.82] | [3.92, 5.64] | [0.159, 0.171] | [0.45, 0.57] | [0.93, 0.96] |
| 0.06  | 0.84  | 3.31  | 0.173 | 0.62  | 0.93 |
|   | [0.81, 0.86] | [2.52, 4.15] | [0.167, 0.179] | [0.57, 0.68] | [0.90, 0.95] |
| 0.08  | 0.86  | 3.22  | 0.179 | 0.66  | 0.91 |
|   | [0.83, 0.88] | [2.40, 4.06] | [0.173, 0.184] | [0.60, 0.73] | [0.88, 0.93] |
| 0.10  | 0.87  | 3.49  | 0.183 | 0.69  | 0.89 |
|   | [0.84, 0.90] | [2.64, 4.41] | [0.177, 0.188] | [0.63, 0.76] | [0.87, 0.92] |
| 0.12  | 0.87  | 3.58  | 0.184 | 0.69  | 0.88 |
|   | [0.84, 0.89] | [2.71, 4.47] | [0.179, 0.189] | [0.62, 0.75] | [0.86, 0.90] |
| 0.14  | 0.85  | 3.99  | 0.180 | 0.67  | 0.87 |
|   | [0.83, 0.88] | [3.16, 4.87] | [0.175, 0.186] | [0.60, 0.73] | [0.85, 0.89] |
| 0.16  | 0.84  | 4.49  | 0.180 | 0.67  | 0.84 |
|   | [0.82, 0.87] | [3.66, 5.36] | [0.174, 0.185] | [0.61, 0.73] | [0.82, 0.87] |
| 0.18  | 0.81  | 5.63  | 0.171 | 0.62  | 0.80 |
|   | [0.79, 0.84] | [4.78, 6.52] | [0.164, 0.177] | [0.56, 0.68] | [0.77, 0.82] |
| 0.20  | 0.80  | 6.32  | 0.166 | 0.60  | 0.75 |
|   | [0.77, 0.82] | [5.45, 7.22] | [0.159, 0.173] | [0.54, 0.66] | [0.72, 0.78] |
| 0.22  | 0.78  | 6.78  | 0.165 | 0.62  | 0.71 |
|   | [0.76, 0.81] | [5.97, 7.64] | [0.158, 0.172] | [0.56, 0.68] | [0.68, 0.74] |
| 0.24  | 0.77  | 7.01  | 0.163 | 0.62  | 0.68 |
|   | [0.75, 0.79] | [6.24, 7.83] | [0.156, 0.169] | [0.57, 0.68] | [0.65, 0.71] |
| 0.26  | 0.74  | 7.97  | 0.158 | 0.60  | 0.62 |
|   | [0.72, 0.76] | [7.25, 8.71] | [0.152, 0.164] | [0.55, 0.66] | [0.59, 0.65] |
| 0.28  | 0.75  | 7.58  | 0.161 | 0.65  | 0.62 |
|   | [0.73, 0.77] | [6.88, 8.32] | [0.154, 0.168] | [0.59, 0.70] | [0.59, 0.65] |
| 0.30  | 0.74  | 7.80  | 0.161 | 0.67  | 0.59 |
|   | [0.72, 0.76] | [7.08, 8.54] | [0.155, 0.168] | [0.62, 0.73] | [0.56, 0.62] |

Table 1: Shapes3d dataset. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
| noise | topo  | conf  | cont  | pos  | acc  |
|-------|-------|-------|-------|------|------|
| 0.00  | 0.71  | 5.54  | 0.153 | 0.45 | 0.95 |
|       | [0.67, 0.74] | [4.73, 6.37] | [0.147, 0.159] | [0.40, 0.50] | [0.94, 0.97] |
| 0.02  | 0.74  | 6.15  | 0.152 | 0.38 | 0.98 |
|       | [0.72, 0.76] | [5.36, 6.91] | [0.146, 0.157] | [0.33, 0.43] | [0.98, 0.98] |
| 0.04  | 0.78  | 5.19  | 0.160 | 0.45 | 0.97 |
|       | [0.76, 0.80] | [4.42, 5.97] | [0.155, 0.165] | [0.39, 0.50] | [0.97, 0.98] |
| 0.06  | 0.79  | 4.36  | 0.164 | 0.52 | 0.92 |
|       | [0.77, 0.82] | [3.61, 4.14] | [0.158, 0.170] | [0.46, 0.58] | [0.90, 0.94] |
| 0.08  | 0.83  | 3.58  | 0.172 | 0.59 | 0.91 |
|       | [0.80, 0.85] | [2.84, 4.32] | [0.166, 0.177] | [0.54, 0.65] | [0.89, 0.94] |
| 0.10  | 0.85  | 3.30  | 0.177 | 0.64 | 0.91 |
|       | [0.82, 0.88] | [2.54, 4.09] | [0.171, 0.183] | [0.58, 0.70] | [0.89, 0.93] |
| 0.12  | 0.86  | 3.14  | 0.182 | 0.70 | 0.90 |
|       | [0.84, 0.89] | [2.39, 3.93] | [0.177, 0.188] | [0.64, 0.76] | [0.87, 0.92] |
| 0.14  | 0.86  | 3.50  | 0.181 | 0.69 | 0.88 |
|       | [0.83, 0.88] | [2.69, 4.31] | [0.175, 0.187] | [0.62, 0.75] | [0.86, 0.90] |
| 0.16  | 0.86  | 3.87  | 0.180 | 0.67 | 0.87 |
|       | [0.83, 0.88] | [3.02, 4.74] | [0.174, 0.186] | [0.60, 0.73] | [0.85, 0.90] |
| 0.18  | 0.82  | 4.73  | 0.171 | 0.63 | 0.82 |
|       | [0.80, 0.85] | [3.93, 5.55] | [0.164, 0.178] | [0.58, 0.69] | [0.79, 0.85] |
| 0.20  | 0.80  | 6.34  | 0.164 | 0.60 | 0.75 |
|       | [0.78, 0.82] | [5.46, 7.22] | [0.157, 0.171] | [0.54, 0.65] | [0.72, 0.78] |
| 0.22  | 0.78  | 7.22  | 0.161 | 0.60 | 0.69 |
|       | [0.76, 0.80] | [6.39, 7.99] | [0.154, 0.168] | [0.54, 0.65] | [0.65, 0.72] |
| 0.24  | 0.76  | 7.66  | 0.157 | 0.62 | 0.62 |
|       | [0.75, 0.78] | [6.98, 8.30] | [0.150, 0.164] | [0.57, 0.66] | [0.59, 0.65] |
| 0.26  | 0.74  | 8.42  | 0.151 | 0.59 | 0.57 |
|       | [0.73, 0.76] | [7.91, 8.88] | [0.144, 0.157] | [0.55, 0.63] | [0.55, 0.60] |
| 0.28  | 0.74  | 8.81  | 0.149 | 0.60 | 0.54 |
|       | [0.73, 0.75] | [8.33, 9.27] | [0.142, 0.156] | [0.57, 0.64] | [0.52, 0.57] |
| 0.30  | 0.73  | 9.94  | 0.149 | 0.64 | 0.51 |
|       | [0.72, 0.74] | [8.47, 9.41] | [0.142, 0.156] | [0.60, 0.67] | [0.49, 0.53] |

Table 2: Obverter dataset. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here *topo* stands for topographic similarity, *conf* for conflict count, *cont* for context independence, *pos* for positional disentanglement and *acc* for accuracy.
D.2 Different number of symbols

This section presents detailed results for the case when the message space has $16 = 4 \times 4$ elements (4-symbol alphabet; see Figure 15 and Table 3) and $64 = 8 \times 8$ elements (8-symbol alphabet; see Figure 16 and Table 4). In the former case, the results are more variable. For topo, this can be seen from a bimodal shape of its distribution and wide confidence intervals, particularly for small to medium values of noise. This makes it statistically hard to distinguish values of topo in this noise range. For larger values of noise (greater than 0.16), topo starts to visibly decline. Similar behavior can be seen for other metrics.

Interestingly, for the latter experiment, with $64 = 8 \times 8$ messages, the story is different. While topographic similarity values for the small noise regime (up to 0.08) do not improve over the baseline value, this behavior changes for medium to large values of noise. In this range, we can observe a visible increase in the topo, peaking at 0.88 with a noise level of 0.24.

![Figure 15: 4 x 4. Top panel: average value of metrics for various noise levels (on shapes3d dataset). The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.](image)

![Figure 16: 8x8. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.](image)
| noise | topo | conf | cont | pos | acc |
|-------|------|------|------|-----|-----|
| 0.00  | 0.74 | 6.45 | 0.154| 0.59| 0.71|
|       | [0.69, 0.78]| [5.30, 7.57]| [0.146, 0.161]| [0.52, 0.67]| [0.66, 0.75]|
| 0.02  | 0.73 | 6.48 | 0.154| 0.57| 0.66|
|       | [0.69, 0.77]| [5.44, 7.48]| [0.148, 0.160]| [0.51, 0.64]| [0.60, 0.72]|
| 0.04  | 0.75 | 5.93 | 0.155| 0.62| 0.65|
|       | [0.71, 0.79]| [4.88, 6.96]| [0.148, 0.162]| [0.56, 0.69]| [0.59, 0.71]|
| 0.06  | 0.76 | 5.94 | 0.156| 0.59| 0.65|
|       | [0.71, 0.80]| [4.92, 6.99]| [0.149, 0.163]| [0.52, 0.66]| [0.59, 0.70]|
| 0.08  | 0.74 | 6.19 | 0.157| 0.61| 0.63|
|       | [0.71, 0.78]| [5.20, 7.19]| [0.151, 0.163]| [0.55, 0.68]| [0.58, 0.68]|
| 0.10  | 0.72 | 7.00 | 0.153| 0.61| 0.60|
|       | [0.69, 0.75]| [6.12, 7.87]| [0.147, 0.159]| [0.55, 0.67]| [0.56, 0.63]|
| 0.12  | 0.75 | 6.90 | 0.159| 0.65| 0.63|
|       | [0.72, 0.78]| [5.95, 7.82]| [0.154, 0.165]| [0.59, 0.71]| [0.60, 0.67]|
| 0.14  | 0.71 | 8.20 | 0.155| 0.60| 0.59|
|       | [0.69, 0.73]| [7.34, 9.02]| [0.149, 0.160]| [0.54, 0.65]| [0.57, 0.62]|
| 0.16  | 0.73 | 8.01 | 0.159| 0.62| 0.62|
|       | [0.71, 0.75]| [7.10, 8.91]| [0.153, 0.165]| [0.56, 0.68]| [0.59, 0.65]|
| 0.18  | 0.72 | 8.18 | 0.153| 0.64| 0.60|
|       | [0.70, 0.74]| [7.34, 9.01]| [0.147, 0.159]| [0.59, 0.70]| [0.57, 0.63]|
| 0.20  | 0.71 | 8.93 | 0.149| 0.63| 0.56|
|       | [0.69, 0.72]| [8.23, 9.63]| [0.143, 0.155]| [0.58, 0.68]| [0.53, 0.59]|
| 0.22  | 0.70 | 9.38 | 0.147| 0.62| 0.54|
|       | [0.69, 0.72]| [8.67, 10.09]| [0.140, 0.153]| [0.57, 0.67]| [0.51, 0.56]|
| 0.24  | 0.69 | 9.86 | 0.147| 0.62| 0.51|
|       | [0.68, 0.70]| [9.23, 10.46]| [0.141, 0.153]| [0.57, 0.67]| [0.49, 0.53]|
| 0.26  | 0.69 | 9.87 | 0.146| 0.64| 0.49|
|       | [0.68, 0.70]| [9.29, 10.45]| [0.140, 0.153]| [0.58, 0.69]| [0.47, 0.52]|
| 0.28  | 0.68 | 10.15| 0.142| 0.66| 0.47|
|       | [0.67, 0.70]| [9.60, 10.72]| [0.136, 0.149]| [0.61, 0.71]| [0.45, 0.49]|
| 0.30  | 0.68 | 10.32| 0.141| 0.68| 0.46|
|       | [0.67, 0.69]| [9.78, 10.86]| [0.135, 0.148]| [0.63, 0.75]| [0.44, 0.48]|

Table 3: 4x4. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
Table 4: 8x8. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95\%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.

| noise | topo | conf | cont | pos | acc |
|-------|------|------|------|-----|-----|
| 0.00  | 0.79 | 2.77 | 0.173| 0.48| 0.98|
|       | [0.77, 0.82] | [2.17, 3.39] | [0.166, 0.178] | [0.42, 0.54] | [0.98, 0.99] |
| 0.02  | 0.74 | 2.99 | 0.160| 0.36| 0.98|
|       | [0.73, 0.76] | [2.43, 3.55] | [0.155, 0.166] | [0.32, 0.40] | [0.98, 0.99] |
| 0.04  | 0.75 | 3.62 | 0.161| 0.34| 0.98|
|       | [0.73, 0.76] | [3.02, 4.22] | [0.156, 0.167] | [0.30, 0.39] | [0.98, 0.99] |
| 0.06  | 0.74 | 3.90 | 0.159| 0.33| 0.98|
|       | [0.73, 0.76] | [3.33, 4.49] | [0.153, 0.165] | [0.28, 0.37] | [0.98, 0.99] |
| 0.08  | 0.77 | 3.94 | 0.165| 0.37| 0.98|
|       | [0.75, 0.79] | [3.33, 4.57] | [0.160, 0.171] | [0.32, 0.42] | [0.98, 0.99] |
| 0.10  | 0.80 | 3.74 | 0.173| 0.43| 0.98|
|       | [0.78, 0.82] | [3.10, 4.38] | [0.167, 0.179] | [0.37, 0.50] | [0.98, 0.99] |
| 0.12  | 0.83 | 3.60 | 0.182| 0.50| 0.98|
|       | [0.80, 0.85] | [2.90, 4.32] | [0.176, 0.189] | [0.43, 0.56] | [0.98, 0.99] |
| 0.14  | 0.84 | 3.82 | 0.186| 0.51| 0.98|
|       | [0.82, 0.86] | [3.09, 4.59] | [0.179, 0.192] | [0.44, 0.58] | [0.98, 0.99] |
| 0.16  | 0.83 | 4.35 | 0.184| 0.49| 0.97|
|       | [0.81, 0.86] | [3.59, 5.11] | [0.177, 0.190] | [0.42, 0.56] | [0.97, 0.99] |
| 0.18  | 0.85 | 4.13 | 0.189| 0.53| 0.97|
|       | [0.83, 0.87] | [3.34, 4.95] | [0.182, 0.195] | [0.46, 0.60] | [0.96, 0.97] |
| 0.20  | 0.86 | 3.97 | 0.189| 0.56| 0.95|
|       | [0.84, 0.88] | [3.24, 4.73] | [0.182, 0.195] | [0.49, 0.63] | [0.94, 0.96] |
| 0.22  | 0.86 | 3.72 | 0.190| 0.59| 0.94|
|       | [0.84, 0.89] | [3.01, 4.48] | [0.183, 0.196] | [0.53, 0.66] | [0.93, 0.96] |
| 0.24  | 0.88 | 3.47 | 0.192| 0.65| 0.93|
|       | [0.85, 0.90] | [2.73, 4.25] | [0.185, 0.198] | [0.58, 0.72] | [0.91, 0.95] |
| 0.26  | 0.87 | 3.84 | 0.190| 0.66| 0.90|
|       | [0.85, 0.89] | [3.11, 4.62] | [0.183, 0.197] | [0.59, 0.72] | [0.87, 0.92] |
| 0.28  | 0.85 | 4.18 | 0.188| 0.66| 0.86|
|       | [0.83, 0.87] | [3.51, 4.89] | [0.181, 0.194] | [0.60, 0.73] | [0.83, 0.88] |
| 0.30  | 0.85 | 4.15 | 0.186| 0.68| 0.84|
|       | [0.82, 0.87] | [3.48, 4.85] | [0.179, 0.193] | [0.62, 0.74] | [0.81, 0.86] |

D.3 Variable noise

In this section, we present detailed results for variable noise experiments. More precisely, at the beginning of training the noise is kept at some initial level $\epsilon_0 \in \{0.0, 0.15\}$ and after $T \in \{200, 700, 1200, 1700, 2200, 2700\}$ warmup network updates it is changed to a value, $\epsilon_T$, and kept there for the rest of the training. The results are presented in Figure 17 and Table 5 ($\epsilon_0 = 0.0, \epsilon_T = 0.15$), Figure 18 and Table 6 ($\epsilon_0 = 0.0, \epsilon_T = 0.1$), and Figure 19 and Table 7 ($\epsilon_0 = 0.15, \epsilon_T = 0.1$).

For $\epsilon_0 = 0.0, \epsilon_T = 0.15$ case, we see that topo increases from 0.85 for $T = 200$, to 0.9 for $T = 2700$, and the transition is reflected both in the density profile as well as the confidence intervals. The effect for $\epsilon_0 = 0.0, \epsilon_T = 0.1$ is weaker and the variance in the results is quite high. There seems to be a negligible effect for the case $\epsilon_0 = 0.15, \epsilon_T = 0.1$. 

25
Figure 17: Variable noise with $\epsilon_0 = 0.0$, $\epsilon_T = 0.15$. Top panel: average value of metrics for various warmup levels ($T$). The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.

Figure 18: Variable noise $\epsilon_0 = 0.0$, $\epsilon_T = 0.1$. Top panel: average value of metrics for various warmup levels ($T$). The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
Figure 19: Variable noise $\epsilon_0 = 0.15$, $\epsilon_T = 0.1$. Top panel: average value of metrics for various warmup levels ($T$). The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.

| warmup | topo    | conf | cont  | pos   | acc   |
|--------|---------|------|-------|-------|-------|
| 200    | 0.85    | 4.06 | 0.180 | 0.67  | 0.85  |
|        | [0.82, 0.87] | [3.25, 4.89] | [0.175, 0.185] | [0.61, 0.73] | [0.83, 0.88] |
| 700    | 0.85    | 4.21 | 0.179 | 0.68  | 0.84  |
|        | [0.82, 0.87] | [3.39, 5.07] | [0.173, 0.184] | [0.62, 0.73] | [0.82, 0.87] |
| 1200   | 0.89    | 3.14 | 0.186 | 0.75  | 0.89  |
|        | [0.86, 0.91] | [2.37, 4.00] | [0.181, 0.191] | [0.69, 0.81] | [0.87, 0.91] |
| 1700   | 0.89    | 3.12 | 0.185 | 0.75  | 0.88  |
|        | [0.86, 0.91] | [2.36, 3.99] | [0.180, 0.190] | [0.68, 0.81] | [0.86, 0.91] |
| 2200   | 0.89    | 3.40 | 0.185 | 0.73  | 0.89  |
|        | [0.86, 0.91] | [2.53, 4.34] | [0.178, 0.190] | [0.65, 0.79] | [0.87, 0.92] |
| 2700   | 0.90    | 2.82 | 0.188 | 0.77  | 0.91  |
|        | [0.88, 0.93] | [2.03, 3.70] | [0.181, 0.193] | [0.71, 0.84] | [0.89, 0.93] |

Table 5: Variable noise with $\epsilon_0 = 0.0$, $\epsilon_T = 0.15$. Results for the metrics for various warmup levels ($T$). Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
Table 6: Variable noise with \( \epsilon_0 = 0.0, \epsilon_T = 0.1 \). Results for the metrics for various warmup levels \( T \). Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.

| warmup | topo  | conf  | cont  | pos   | acc  |
|--------|-------|-------|-------|-------|------|
| 200    | 0.87  | 3.28  | 0.182 | 0.69  | 0.90 |
|        | [0.84, 0.90] | [2.47, 4.12] | [0.176, 0.187] | [0.63, 0.75] | [0.88, 0.92] |
| 700    | 0.86  | 3.35  | 0.179 | 0.68  | 0.90 |
|        | [0.84, 0.89] | [2.56, 4.17] | [0.173, 0.184] | [0.62, 0.73] | [0.87, 0.92] |
| 1200   | 0.88  | 2.89  | 0.184 | 0.72  | 0.91 |
|        | [0.86, 0.91] | [2.12, 3.70] | [0.179, 0.189] | [0.66, 0.78] | [0.88, 0.93] |
| 1700   | 0.88  | 3.14  | 0.181 | 0.70  | 0.91 |
|        | [0.85, 0.90] | [2.28, 4.04] | [0.175, 0.187] | [0.65, 0.76] | [0.89, 0.93] |
| 2200   | 0.89  | 2.69  | 0.183 | 0.74  | 0.92 |
|        | [0.86, 0.92] | [1.85, 3.54] | [0.177, 0.189] | [0.68, 0.80] | [0.90, 0.94] |
| 2700   | 0.89  | 2.56  | 0.183 | 0.74  | 0.92 |
|        | [0.86, 0.92] | [1.78, 3.40] | [0.176, 0.189] | [0.68, 0.80] | [0.90, 0.94] |

Table 7: Variable noise with \( \epsilon_0 = 0.15, \epsilon_T = 0.1 \). Results for the metrics for various warmup levels \( T \). Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.

| warmup | topo  | conf  | cont  | pos   | acc  |
|--------|-------|-------|-------|-------|------|
| 200    | 0.86  | 3.73  | 0.181 | 0.67  | 0.88 |
|        | [0.83, 0.88] | [2.85, 4.63] | [0.175, 0.186] | [0.61, 0.73] | [0.85, 0.90] |
| 700    | 0.85  | 3.91  | 0.178 | 0.65  | 0.88 |
|        | [0.82, 0.87] | [3.04, 4.76] | [0.172, 0.183] | [0.59, 0.71] | [0.85, 0.90] |
| 1200   | 0.85  | 3.63  | 0.180 | 0.66  | 0.88 |
|        | [0.83, 0.88] | [2.81, 4.46] | [0.175, 0.185] | [0.59, 0.72] | [0.86, 0.90] |
| 1700   | 0.85  | 3.77  | 0.179 | 0.65  | 0.88 |
|        | [0.82, 0.88] | [2.90, 4.61] | [0.174, 0.184] | [0.59, 0.71] | [0.86, 0.90] |
| 2200   | 0.85  | 3.86  | 0.179 | 0.64  | 0.88 |
|        | [0.82, 0.87] | [2.99, 4.71] | [0.173, 0.184] | [0.58, 0.70] | [0.85, 0.90] |
| 2700   | 0.84  | 3.94  | 0.177 | 0.62  | 0.88 |
|        | [0.81, 0.87] | [3.08, 4.76] | [0.171, 0.183] | [0.56, 0.69] | [0.86, 0.90] |
D.4 Sensitivity with respect to small architecture changes

This section provides details for experiments aiming to check the impact of small architecture change in CNN (Figure 20 Table 8) and the dense layers in the agents’ network (Figure 21 Table 9). In the former experiment, we change the number of filters in the sender’s CNN architecture from two layers with 8 filters, to two layers with 16 filters. This results in the slight change of topographic similarity profile with the highest average value of 0.86 for noise range 0.14 significantly outperforming the zero-noise case (0.79). For the second experiment, we added a dense layer (with 64 neurons) to the receiver’s architecture. This again shows improvement of compositionality due to noise, although the noise in the range [0.08, 0.16] performs roughly the same.

Figure 20: Bigger CNN. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.

Figure 21: Bigger dense layer. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
| noise | topo | conf | cont | pos | acc |
|-------|------|------|------|-----|-----|
| 0.00  | 0.79 | 4.23 | 0.168 | 0.56 | 0.98 |
|       | [0.76, 0.82] | [3.34, 5.11] | [0.160, 0.174] | [0.49, 0.62] | [0.97, 0.98] |
| 0.02  | 0.81 | 4.47 | 0.166 | 0.51 | 0.98 |
|       | [0.78, 0.83] | [3.63, 5.34] | [0.160, 0.172] | [0.45, 0.57] | [0.98, 0.98] |
| 0.04  | 0.80 | 4.53 | 0.166 | 0.53 | 0.95 |
|       | [0.78, 0.83] | [3.66, 5.42] | [0.160, 0.172] | [0.47, 0.60] | [0.93, 0.96] |
| 0.06  | 0.84 | 3.35 | 0.175 | 0.63 | 0.92 |
|       | [0.81, 0.87] | [2.56, 4.22] | [0.169, 0.181] | [0.57, 0.69] | [0.89, 0.94] |
| 0.08  | 0.85 | 3.32 | 0.178 | 0.65 | 0.90 |
|       | [0.83, 0.88] | [2.52, 4.18] | [0.172, 0.184] | [0.58, 0.71] | [0.88, 0.93] |
| 0.10  | 0.85 | 3.86 | 0.178 | 0.63 | 0.89 |
|       | [0.83, 0.88] | [2.99, 4.76] | [0.172, 0.183] | [0.57, 0.70] | [0.86, 0.91] |
| 0.12  | 0.86 | 4.06 | 0.179 | 0.65 | 0.88 |
|       | [0.83, 0.89] | [3.13, 4.95] | [0.173, 0.185] | [0.58, 0.72] | [0.85, 0.90] |
| 0.14  | 0.86 | 3.90 | 0.181 | 0.67 | 0.88 |
|       | [0.84, 0.89] | [3.00, 4.81] | [0.176, 0.186] | [0.61, 0.73] | [0.86, 0.90] |
| 0.16  | 0.84 | 4.64 | 0.176 | 0.63 | 0.85 |
|       | [0.82, 0.87] | [3.70, 5.59] | [0.170, 0.182] | [0.56, 0.70] | [0.83, 0.87] |
| 0.18  | 0.83 | 5.17 | 0.174 | 0.62 | 0.82 |
|       | [0.80, 0.85] | [4.22, 6.08] | [0.168, 0.180] | [0.56, 0.69] | [0.80, 0.85] |
| 0.20  | 0.80 | 6.26 | 0.168 | 0.59 | 0.77 |
|       | [0.78, 0.83] | [5.32, 7.17] | [0.162, 0.174] | [0.53, 0.65] | [0.75, 0.80] |
| 0.22  | 0.80 | 6.04 | 0.167 | 0.65 | 0.74 |
|       | [0.78, 0.82] | [5.21, 6.85] | [0.161, 0.173] | [0.59, 0.70] | [0.72, 0.77] |
| 0.24  | 0.80 | 6.26 | 0.166 | 0.65 | 0.72 |
|       | [0.78, 0.82] | [5.46, 7.04] | [0.160, 0.172] | [0.59, 0.71] | [0.69, 0.75] |
| 0.26  | 0.77 | 6.95 | 0.161 | 0.66 | 0.66 |
|       | [0.75, 0.79] | [6.19, 7.66] | [0.155, 0.168] | [0.61, 0.72] | [0.62, 0.69] |
| 0.28  | 0.76 | 7.65 | 0.161 | 0.66 | 0.62 |
|       | [0.74, 0.78] | [6.91, 8.38] | [0.154, 0.167] | [0.60, 0.72] | [0.59, 0.66] |
| 0.30  | 0.75 | 7.81 | 0.160 | 0.66 | 0.61 |
|       | [0.73, 0.77] | [7.08, 8.55] | [0.153, 0.166] | [0.60, 0.73] | [0.57, 0.64] |

Table 8: Bigger CNN. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
| noise | topo | conf | cont | pos  | acc  |
|-------|------|------|------|------|------|
| 0.00  | 0.76 | 4.42 | 0.162| 0.56 | 0.93 |
|       | [0.72, 0.80] | [3.45, 5.38] | [0.155, 0.169] | [0.49, 0.62] | [0.91, 0.95] |
| 0.02  | 0.80 | 4.10 | 0.167| 0.55 | 0.96 |
|       | [0.77, 0.83] | [3.30, 4.92] | [0.160, 0.173] | [0.50, 0.61] | [0.95, 0.98] |
| 0.04  | 0.78 | 4.65 | 0.167| 0.54 | 0.90 |
|       | [0.76, 0.81] | [3.83, 5.42] | [0.161, 0.173] | [0.47, 0.59] | [0.88, 0.93] |
| 0.06  | 0.82 | 3.62 | 0.173| 0.62 | 0.86 |
|       | [0.79, 0.85] | [2.80, 4.48] | [0.167, 0.179] | [0.56, 0.68] | [0.83, 0.90] |
| 0.08  | 0.86 | 3.14 | 0.180| 0.69 | 0.87 |
|       | [0.83, 0.88] | [2.33, 3.97] | [0.174, 0.185] | [0.63, 0.75] | [0.84, 0.90] |
| 0.10  | 0.85 | 3.27 | 0.179| 0.69 | 0.86 |
|       | [0.83, 0.88] | [2.48, 4.09] | [0.173, 0.185] | [0.63, 0.75] | [0.83, 0.88] |
| 0.12  | 0.86 | 3.56 | 0.180| 0.68 | 0.87 |
|       | [0.83, 0.88] | [2.76, 4.42] | [0.175, 0.186] | [0.62, 0.74] | [0.84, 0.89] |
| 0.14  | 0.86 | 3.95 | 0.180| 0.67 | 0.86 |
|       | [0.83, 0.88] | [3.14, 4.87] | [0.174, 0.186] | [0.61, 0.74] | [0.84, 0.89] |
| 0.16  | 0.85 | 4.31 | 0.177| 0.67 | 0.84 |
|       | [0.82, 0.87] | [3.46, 5.21] | [0.171, 0.183] | [0.61, 0.73] | [0.81, 0.86] |
| 0.18  | 0.82 | 5.42 | 0.173| 0.65 | 0.79 |
|       | [0.80, 0.85] | [4.46, 6.36] | [0.167, 0.179] | [0.58, 0.71] | [0.76, 0.83] |
| 0.20  | 0.79 | 6.25 | 0.169| 0.62 | 0.73 |
|       | [0.76, 0.81] | [5.33, 7.14] | [0.163, 0.175] | [0.57, 0.68] | [0.69, 0.76] |
| 0.22  | 0.77 | 7.04 | 0.163| 0.61 | 0.67 |
|       | [0.74, 0.79] | [6.17, 7.85] | [0.157, 0.170] | [0.56, 0.67] | [0.64, 0.71] |
| 0.24  | 0.74 | 8.14 | 0.159| 0.59 | 0.62 |
|       | [0.72, 0.76] | [7.36, 8.87] | [0.153, 0.166] | [0.54, 0.64] | [0.59, 0.65] |
| 0.26  | 0.74 | 8.08 | 0.159| 0.64 | 0.60 |
|       | [0.72, 0.76] | [7.33, 8.82] | [0.153, 0.165] | [0.59, 0.69] | [0.57, 0.63] |
| 0.28  | 0.73 | 8.58 | 0.158| 0.63 | 0.57 |
|       | [0.71, 0.75] | [7.90, 9.24] | [0.152, 0.164] | [0.57, 0.68] | [0.54, 0.59] |
| 0.30  | 0.71 | 9.21 | 0.154| 0.61 | 0.53 |
|       | [0.69, 0.73] | [8.58, 9.85] | [0.147, 0.160] | [0.56, 0.67] | [0.51, 0.56] |

Table 9: Bigger dense layer. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
D.5 Sensitivity to noisy channel implementation

In this section, we provide details for experiments with alternative noisy channel implementation, see Figure 22 and Table 10. Notice that the range of noise values is different from the main experiment (smaller by the order of magnitude). The results suggest that the small values of noise can help, while the larger noise levels lead to a decline in compositionality.

Figure 22: Alternative noise design. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
| noise | topo  | conf  | cont  | pos  | acc  |
|-------|-------|-------|-------|------|------|
| 0.000 | 0.79  | 4.13  | 0.167 | 0.56 | 0.97 |
|       | [0.75, 0.82] | [3.31, 4.95] | [0.160, 0.173] | [0.50, 0.62] | [0.96, 0.98] |
| 0.001 | 0.79  | 3.77  | 0.168 | 0.59 | 0.97 |
|       | [0.76, 0.82] | [3.00, 4.57] | [0.162, 0.174] | [0.53, 0.64] | [0.96, 0.98] |
| 0.002 | 0.80  | 3.51  | 0.170 | 0.60 | 0.96 |
|       | [0.77, 0.83] | [2.77, 4.29] | [0.164, 0.176] | [0.55, 0.66] | [0.95, 0.97] |
| 0.003 | 0.79  | 3.52  | 0.169 | 0.61 | 0.95 |
|       | [0.76, 0.82] | [2.77, 4.31] | [0.163, 0.175] | [0.55, 0.66] | [0.94, 0.96] |
| 0.004 | 0.77  | 3.80  | 0.166 | 0.58 | 0.94 |
|       | [0.74, 0.81] | [2.97, 4.66] | [0.160, 0.172] | [0.53, 0.64] | [0.92, 0.95] |
| 0.005 | 0.77  | 3.81  | 0.165 | 0.58 | 0.93 |
|       | [0.73, 0.80] | [2.99, 4.65] | [0.158, 0.171] | [0.53, 0.63] | [0.91, 0.95] |
| 0.006 | 0.75  | 4.21  | 0.163 | 0.56 | 0.91 |
|       | [0.72, 0.79] | [3.38, 5.11] | [0.156, 0.170] | [0.50, 0.61] | [0.89, 0.93] |
| 0.007 | 0.76  | 4.05  | 0.163 | 0.56 | 0.91 |
|       | [0.72, 0.79] | [3.26, 4.89] | [0.156, 0.169] | [0.51, 0.62] | [0.89, 0.92] |
| 0.008 | 0.74  | 4.48  | 0.158 | 0.54 | 0.89 |
|       | [0.70, 0.77] | [3.65, 5.31] | [0.152, 0.164] | [0.48, 0.59] | [0.87, 0.91] |
| 0.009 | 0.73  | 4.68  | 0.158 | 0.52 | 0.88 |
|       | [0.70, 0.77] | [3.81, 5.55] | [0.151, 0.164] | [0.47, 0.58] | [0.85, 0.90] |
| 0.010 | 0.72  | 4.70  | 0.156 | 0.52 | 0.87 |
|       | [0.69, 0.76] | [3.87, 5.53] | [0.150, 0.162] | [0.47, 0.58] | [0.85, 0.89] |
| 0.011 | 0.70  | 5.43  | 0.153 | 0.48 | 0.85 |
|       | [0.67, 0.74] | [4.59, 6.30] | [0.147, 0.159] | [0.43, 0.54] | [0.83, 0.88] |
| 0.012 | 0.69  | 5.87  | 0.150 | 0.46 | 0.84 |
|       | [0.65, 0.72] | [5.00, 6.76] | [0.144, 0.157] | [0.40, 0.51] | [0.81, 0.86] |
| 0.013 | 0.67  | 6.29  | 0.146 | 0.43 | 0.83 |
|       | [0.63, 0.70] | [5.42, 7.19] | [0.140, 0.152] | [0.37, 0.48] | [0.80, 0.85] |
| 0.014 | 0.65  | 6.23  | 0.144 | 0.43 | 0.81 |
|       | [0.62, 0.68] | [5.47, 7.03] | [0.138, 0.151] | [0.38, 0.48] | [0.79, 0.83] |
| 0.015 | 0.63  | 6.91  | 0.145 | 0.39 | 0.80 |
|       | [0.61, 0.66] | [6.13, 7.72] | [0.139, 0.150] | [0.34, 0.44] | [0.78, 0.82] |

Table 10: Alternative noise design. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
D.6 Longer message

Here we provide details for longer message experiments, see Figure 23, Table 11 and Figure 24, Table 12 for message lengths three and four, respectively. The setup expands upon the main experiment by including floor color for the former and, additionally, wall color for the latter. The overall levels of compositionality metrics decline when compared with the main experiment, however, the general picture that noise improves compositionality remains intact.

Figure 23: Message length equals 3. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, pos for positional disentanglement and acc for accuracy.

Figure 24: Message length equals 4. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, pos for positional disentanglement and acc for accuracy.

---

11 We took the floor color feature to take values 0, 1, 2, 3 (out of 10 possible). The resulting dataset contains 76800 images.

12 We took the wall color feature to take values 0, 1, 2, 3 (out of 10 possible). The resulting dataset contains 30720 images.
| noise | topo    | cont    | pos    | acc    |
|-------|---------|---------|--------|--------|
| 0.00  | 0.76    | 0.069   | 0.49   | 0.89   |
|       | [0.73, 0.78] | [0.067, 0.072] | [0.44, 0.54] | [0.85, 0.92] |
| 0.02  | 0.76    | 0.069   | 0.45   | 0.93   |
|       | [0.74, 0.78] | [0.067, 0.071] | [0.41, 0.50] | [0.91, 0.95] |
| 0.04  | 0.77    | 0.069   | 0.48   | 0.91   |
|       | [0.75, 0.79] | [0.067, 0.071] | [0.43, 0.53] | [0.89, 0.93] |
| 0.06  | 0.79    | 0.072   | 0.49   | 0.84   |
|       | [0.77, 0.81] | [0.070, 0.074] | [0.44, 0.54] | [0.81, 0.87] |
| 0.08  | 0.80    | 0.073   | 0.51   | 0.81   |
|       | [0.78, 0.82] | [0.071, 0.074] | [0.46, 0.56] | [0.77, 0.84] |
| 0.10  | 0.80    | 0.073   | 0.51   | 0.76   |
|       | [0.78, 0.82] | [0.071, 0.075] | [0.46, 0.56] | [0.73, 0.79] |
| 0.12  | 0.79    | 0.073   | 0.51   | 0.74   |
|       | [0.77, 0.81] | [0.071, 0.075] | [0.46, 0.56] | [0.71, 0.78] |
| 0.14  | 0.76    | 0.070   | 0.46   | 0.65   |
|       | [0.74, 0.78] | [0.068, 0.072] | [0.41, 0.51] | [0.62, 0.68] |
| 0.16  | 0.76    | 0.069   | 0.46   | 0.59   |
|       | [0.74, 0.77] | [0.067, 0.071] | [0.42, 0.51] | [0.56, 0.63] |
| 0.18  | 0.74    | 0.069   | 0.45   | 0.52   |
|       | [0.73, 0.76] | [0.067, 0.071] | [0.41, 0.49] | [0.49, 0.56] |
| 0.20  | 0.74    | 0.069   | 0.45   | 0.49   |
|       | [0.73, 0.76] | [0.067, 0.071] | [0.40, 0.49] | [0.46, 0.53] |
| 0.22  | 0.73    | 0.066   | 0.45   | 0.44   |
|       | [0.72, 0.74] | [0.064, 0.068] | [0.40, 0.49] | [0.41, 0.47] |
| 0.24  | 0.72    | 0.065   | 0.45   | 0.41   |
|       | [0.71, 0.73] | [0.063, 0.067] | [0.40, 0.49] | [0.38, 0.43] |
| 0.26  | 0.72    | 0.065   | 0.45   | 0.37   |
|       | [0.71, 0.73] | [0.064, 0.067] | [0.41, 0.50] | [0.35, 0.40] |
| 0.28  | 0.71    | 0.065   | 0.46   | 0.35   |
|       | [0.70, 0.72] | [0.064, 0.067] | [0.42, 0.51] | [0.33, 0.37] |
| 0.30  | 0.71    | 0.064   | 0.46   | 0.33   |
|       | [0.70, 0.72] | [0.063, 0.066] | [0.42, 0.51] | [0.31, 0.36] |

Table 11: Message length equals 3. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
Table 12: Message length equals 4. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.

| noise | topo   | cont   | pos   | acc   |
|-------|--------|--------|-------|-------|
| 0.00  | 0.68   | 0.038  | 0.33  | 0.71  |
|       | [0.65, 0.71] | [0.036, 0.039] | [0.29, 0.38] | [0.66, 0.76] |  
| 0.02  | 0.74   | 0.040  | 0.34  | 0.83  |
|       | [0.72, 0.76] | [0.039, 0.040] | [0.30, 0.39] | [0.79, 0.87] |  
| 0.04  | 0.73   | 0.040  | 0.34  | 0.78  |
|       | [0.71, 0.75] | [0.039, 0.040] | [0.30, 0.39] | [0.74, 0.82] |  
| 0.06  | 0.74   | 0.040  | 0.34  | 0.72  |
|       | [0.73, 0.76] | [0.039, 0.041] | [0.30, 0.39] | [0.69, 0.76] |  
| 0.08  | 0.73   | 0.040  | 0.34  | 0.65  |
|       | [0.72, 0.75] | [0.039, 0.040] | [0.30, 0.39] | [0.61, 0.68] |  
| 0.10  | 0.72   | 0.039  | 0.34  | 0.57  |
|       | [0.70, 0.73] | [0.039, 0.040] | [0.30, 0.38] | [0.54, 0.60] |  
| 0.12  | 0.71   | 0.039  | 0.33  | 0.49  |
|       | [0.69, 0.72] | [0.038, 0.039] | [0.29, 0.37] | [0.47, 0.52] |  
| 0.14  | 0.70   | 0.039  | 0.32  | 0.41  |
|       | [0.69, 0.72] | [0.038, 0.039] | [0.28, 0.36] | [0.38, 0.45] |  
| 0.16  | 0.69   | 0.038  | 0.31  | 0.37  |
|       | [0.68, 0.71] | [0.038, 0.039] | [0.27, 0.35] | [0.34, 0.40] |  
| 0.18  | 0.70   | 0.039  | 0.30  | 0.34  |
|       | [0.69, 0.71] | [0.038, 0.039] | [0.27, 0.34] | [0.31, 0.37] |  
| 0.20  | 0.69   | 0.038  | 0.30  | 0.30  |
|       | [0.67, 0.70] | [0.037, 0.039] | [0.26, 0.34] | [0.27, 0.32] |  
| 0.22  | 0.68   | 0.038  | 0.31  | 0.27  |
|       | [0.67, 0.69] | [0.037, 0.039] | [0.28, 0.35] | [0.25, 0.29] |  
| 0.24  | 0.68   | 0.038  | 0.32  | 0.24  |
|       | [0.67, 0.69] | [0.037, 0.038] | [0.28, 0.36] | [0.22, 0.26] |  
| 0.26  | 0.67   | 0.038  | 0.32  | 0.22  |
|       | [0.66, 0.68] | [0.037, 0.038] | [0.28, 0.36] | [0.21, 0.24] |  
| 0.28  | 0.66   | 0.037  | 0.32  | 0.21  |
|       | [0.66, 0.67] | [0.037, 0.038] | [0.28, 0.35] | [0.20, 0.23] |  
| 0.30  | 0.67   | 0.037  | 0.33  | 0.20  |
|       | [0.66, 0.68] | [0.036, 0.038] | [0.29, 0.37] | [0.18, 0.21] |  

Table 12: Message length equals 4. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
D.7 Visual priors

In this section we give details for visual priors experiments, see Figure 25, Table 13 (tile 32), Figure 26, Table 14 (tile 16), and Figure 27, Table 15 (tile 8). The transition from coarser to finer tiles has a significant impact both on the metric’s profiles and on their overall levels. The characteristic peak for some positive noise levels is still present.

We complement the picture with an analysis of the interplay between variable noise and accuracy, see Figure 28 (tile 32) and Figure 29 (tile 16). It shows that the overall metrics level, conditioned on seeds with high accuracy, increases significantly. Notice, however, that the number of experiments with high accuracy decrease as the threshold increases. In particular, we did not include visualization for a tile size 8, since there were too few experiments exceeding the accuracy threshold of 0.80.

![Figure 25: Scramble with tile 32. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.](image)

![Figure 26: Scramble with tile 16. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.](image)
Figure 27: Scramble with tile 8. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.

Figure 28: Tile size 32. Top left: The values of topo computed for each noise level (hues) and on seeds exceeding a certain accuracy threshold (x-axis). Vertical dashed lines represent three cross-sections, visualized in the bottom panel. Top right: Similar to the left panel, but instead of topo we visualize the number of seeds with accuracy at least as a given threshold (x-axis). Vertical dashed lines represent three cross-sections, visualized in the bottom panel. Bottom: Each of the plots represents a cross-section of the plots in the top panel, taken at points 0.80, 0.85, and 0.90, respectively. On the left axis of each figure is the range of topo, whereas on the right axis is the number of seeds with accuracy exceed the corresponding level. On the x-axis are the noise levels. The scatter plot with 95%-confidence intervals represents the values of topo. The gray dashed line represents the number of seeds with accuracy exceeding a given threshold, for each of the noise levels.
Figure 29: Tile size 16. Top left: The values of topo computed for each noise level (hues) and on seeds exceeding a certain accuracy threshold (x-axis). Vertical dashed lines represent three cross-sections, visualized in the bottom panel. Top right: Similar to the left panel, but instead of topo we visualize the number of seeds with accuracy at least as a given threshold (x-axis). Vertical dashed lines represent three cross-sections, visualized in the bottom panel. Bottom: Each of the plots represents a cross-section of the plots in the top panel, taken at points 0.80, 0.85, and 0.90, respectively. On the left axis of each figure is the range of topo, whereas on the right axis is the number of seeds with accuracy exceed the corresponding level. On the x-axis are the noise levels. The scatter plot with 95%-confidence intervals represents the values of topo. The gray dashed line represents the number of seeds with accuracy exceeding a given threshold, for each of the noise levels.
Table 13: Scramble with tile 32. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
| noise | topo | conf | cont | pos | acc |
|-------|------|------|------|-----|-----|
| 0.00  | 0.67 | 5.32 | 0.146 | 0.49 | 0.79 |
|       | [0.64, 0.71] | [4.36, 6.26] | [0.139, 0.153] | [0.43, 0.55] | [0.77, 0.82] |
| 0.02  | 0.71 | 5.97 | 0.150 | 0.44 | 0.84 |
|       | [0.67, 0.74] | [5.01, 6.89] | [0.144, 0.157] | [0.39, 0.50] | [0.81, 0.86] |
| 0.04  | 0.69 | 6.87 | 0.147 | 0.41 | 0.77 |
|       | [0.67, 0.71] | [6.07, 7.66] | [0.141, 0.152] | [0.37, 0.46] | [0.75, 0.79] |
| 0.06  | 0.68 | 7.79 | 0.147 | 0.39 | 0.70 |
|       | [0.67, 0.70] | [7.13, 8.43] | [0.141, 0.152] | [0.35, 0.43] | [0.69, 0.72] |
| 0.08  | 0.67 | 8.05 | 0.144 | 0.39 | 0.64 |
|       | [0.66, 0.69] | [7.50, 8.60] | [0.139, 0.149] | [0.36, 0.42] | [0.62, 0.66] |
| 0.10  | 0.66 | 8.81 | 0.142 | 0.40 | 0.60 |
|       | [0.65, 0.68] | [8.27, 9.35] | [0.137, 0.148] | [0.36, 0.44] | [0.58, 0.62] |
| 0.12  | 0.65 | 9.34 | 0.138 | 0.41 | 0.57 |
|       | [0.64, 0.67] | [8.75, 9.91] | [0.132, 0.144] | [0.37, 0.45] | [0.54, 0.59] |
| 0.14  | 0.64 | 10.30 | 0.136 | 0.43 | 0.52 |
|       | [0.62, 0.65] | [9.76, 10.82] | [0.130, 0.142] | [0.39, 0.49] | [0.50, 0.55] |
| 0.16  | 0.63 | 10.54 | 0.137 | 0.42 | 0.50 |
|       | [0.62, 0.64] | [10.03, 11.04] | [0.131, 0.142] | [0.38, 0.47] | [0.48, 0.53] |
| 0.18  | 0.62 | 10.95 | 0.138 | 0.44 | 0.48 |
|       | [0.61, 0.64] | [10.47, 11.43] | [0.132, 0.145] | [0.39, 0.49] | [0.45, 0.50] |
| 0.20  | 0.62 | 11.55 | 0.136 | 0.45 | 0.44 |
|       | [0.60, 0.63] | [11.10, 11.99] | [0.131, 0.142] | [0.40, 0.50] | [0.42, 0.47] |
| 0.22  | 0.61 | 11.95 | 0.133 | 0.47 | 0.42 |
|       | [0.59, 0.62] | [11.47, 12.43] | [0.126, 0.139] | [0.41, 0.53] | [0.40, 0.45] |
| 0.24  | 0.59 | 12.38 | 0.131 | 0.47 | 0.40 |
|       | [0.58, 0.61] | [11.94, 12.84] | [0.125, 0.138] | [0.41, 0.53] | [0.38, 0.42] |
| 0.26  | 0.58 | 12.67 | 0.130 | 0.48 | 0.38 |
|       | [0.57, 0.60] | [12.26, 13.09] | [0.124, 0.136] | [0.42, 0.54] | [0.36, 0.40] |
| 0.28  | 0.58 | 12.97 | 0.132 | 0.49 | 0.36 |
|       | [0.57, 0.59] | [12.55, 13.41] | [0.126, 0.139] | [0.43, 0.55] | [0.34, 0.39] |
| 0.30  | 0.58 | 13.25 | 0.129 | 0.49 | 0.35 |
|       | [0.56, 0.59] | [12.79, 13.73] | [0.123, 0.136] | [0.43, 0.55] | [0.33, 0.37] |

Table 14: Scramble with tile 16. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
| noise | topo    | conf    | cont    | pos    | acc    |
|-------|---------|---------|---------|--------|--------|
| 0.00  | 0.59    | 8.51    | 0.134   | 0.44   | 0.53   |
|       | [0.57, 0.60] | [7.86, 9.13] | [0.130, 0.139] | [0.40, 0.47] | [0.52, 0.54] |
| 0.02  | 0.64    | 8.48    | 0.144   | 0.54   | 0.49   |
|       | [0.63, 0.64] | [8.07, 8.92] | [0.140, 0.149] | [0.52, 0.56] | [0.48, 0.50] |
| 0.04  | 0.65    | 9.82    | 0.138   | 0.70   | 0.38   |
|       | [0.65, 0.66] | [9.46, 10.18] | [0.132, 0.144] | [0.66, 0.73] | [0.37, 0.40] |
| 0.06  | 0.68    | 11.33   | 0.130   | 0.90   | 0.29   |
|       | [0.67, 0.68] | [11.04, 11.58] | [0.123, 0.138] | [0.87, 0.92] | [0.27, 0.30] |
| 0.08  | 0.68    | 11.86   | 0.127   | 0.96   | 0.26   |
|       | [0.68, 0.69] | [11.73, 11.96] | [0.119, 0.135] | [0.94, 0.97] | [0.25, 0.27] |
| 0.10  | 0.68    | 11.95   | 0.126   | 0.97   | 0.25   |
|       | [0.68, 0.69] | [11.87, 12.00] | [0.118, 0.135] | [0.96, 0.98] | [0.24, 0.25] |
| 0.12  | 0.68    | 12.04   | 0.124   | 0.97   | 0.25   |
|       | [0.67, 0.68] | [12.00, 12.12] | [0.116, 0.132] | [0.96, 0.98] | [0.24, 0.25] |
| 0.14  | 0.67    | 12.00   | 0.128   | 0.97   | 0.25   |
|       | [0.67, 0.68] | [12.00, 12.00] | [0.121, 0.136] | [0.96, 0.98] | [0.24, 0.25] |
| 0.16  | 0.66    | 12.03   | 0.123   | 0.98   | 0.25   |
|       | [0.65, 0.67] | [11.98, 12.12] | [0.115, 0.132] | [0.97, 0.98] | [0.24, 0.25] |
| 0.18  | 0.63    | 12.36   | 0.125   | 0.98   | 0.24   |
|       | [0.62, 0.65] | [12.16, 12.60] | [0.118, 0.133] | [0.97, 0.98] | [0.24, 0.25] |
| 0.20  | 0.61    | 12.88   | 0.121   | 0.98   | 0.24   |
|       | [0.60, 0.62] | [12.56, 13.24] | [0.114, 0.129] | [0.97, 0.98] | [0.23, 0.24] |
| 0.22  | 0.59    | 13.24   | 0.119   | 0.98   | 0.23   |
|       | [0.58, 0.61] | [12.88, 13.64] | [0.112, 0.127] | [0.98, 0.98] | [0.22, 0.24] |
| 0.24  | 0.57    | 13.93   | 0.120   | 0.98   | 0.22   |
|       | [0.56, 0.59] | [13.52, 14.34] | [0.113, 0.127] | [0.97, 0.98] | [0.21, 0.23] |
| 0.26  | 0.55    | 14.65   | 0.114   | 0.98   | 0.21   |
|       | [0.53, 0.56] | [14.16, 15.13] | [0.107, 0.121] | [0.97, 0.98] | [0.20, 0.22] |
| 0.28  | 0.54    | 14.92   | 0.111   | 0.98   | 0.20   |
|       | [0.52, 0.55] | [14.44, 15.44] | [0.104, 0.117] | [0.98, 0.98] | [0.20, 0.21] |
| 0.30  | 0.53    | 15.08   | 0.111   | 0.98   | 0.20   |
|       | [0.52, 0.55] | [14.60, 15.56] | [0.105, 0.118] | [0.98, 0.98] | [0.19, 0.21] |

Table 15: Scramble with tile 8. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
D.8 Scrambled features

In these experiments, we aimed to understand if the architecture of the output is important. Our architecture has a separate output for 'color' ($F_1$) and for 'shape' ($F_2$) (for the architecture details, see Figure 8). We permute these features as follows. Let $k : F_1 \times F_2 \mapsto \{0, \ldots, |F_1 \times F_2| - 1\} =: \tilde{F}$ be any bijection from the color features and the shape features. Let $\pi : \tilde{F} \mapsto \tilde{F}$ be a random permutation, sampled at the beginning of the experiment. We assign new features via mapping: $k^{-1}(\pi(k((f_c, f_s))))$. Clearly, these are no longer colors and shapes (unless $\pi$ is an identity). However, they are still factorized along two dimensions and we still use the factorized output. We can see that behavior is similar to the 'standard' features (high levels of compositionality with the characteristic extremum point). This poses strong evidence that output architecture is a strong inductive bias for compositionality.

Figure 30: Scrambled features. Top panel: average value of metrics for various noise levels. The shaded area corresponds to bootstrapped 95%-confidence intervals for this estimator. Bottom panel: kernel density estimators for metrics and noise levels across seeds. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
| noise | topo | conf | cont | pos | acc  |
|-------|------|------|------|-----|------|
|       |      |      |      |     | 0.18 |
| 0.00  | 0.59 | 8.72 | 0.129 | 0.24 | 0.97 |
|       | [0.57, 0.61] | [8.17, 9.26] | [0.125, 0.133] | [0.20, 0.27] | [0.96, 0.98] |
| 0.02  | 0.64 | 8.21 | 0.138 | 0.25 | 0.98 |
|       | [0.63, 0.66] | [7.65, 8.73] | [0.133, 0.142] | [0.21, 0.28] | [0.98, 0.98] |
| 0.04  | 0.68 | 7.07 | 0.145 | 0.32 | 0.97 |
|       | [0.66, 0.70] | [6.39, 7.75] | [0.140, 0.151] | [0.28, 0.37] | [0.97, 0.98] |
| 0.06  | 0.72 | 5.88 | 0.153 | 0.42 | 0.91 |
|       | [0.70, 0.75] | [5.18, 6.58] | [0.147, 0.158] | [0.37, 0.47] | [0.89, 0.93] |
| 0.08  | 0.79 | 4.48 | 0.166 | 0.53 | 0.90 |
|       | [0.76, 0.82] | [3.69, 5.24] | [0.161, 0.172] | [0.47, 0.59] | [0.88, 0.92] |
| 0.10  | 0.85 | 3.40 | 0.179 | 0.65 | 0.88 |
|       | [0.82, 0.88] | [2.58, 4.21] | [0.173, 0.185] | [0.59, 0.72] | [0.85, 0.91] |
| 0.12  | 0.87 | 2.93 | 0.182 | 0.69 | 0.89 |
|       | [0.84, 0.89] | [2.21, 3.69] | [0.176, 0.187] | [0.63, 0.75] | [0.87, 0.91] |
| 0.14  | 0.87 | 3.57 | 0.182 | 0.68 | 0.88 |
|       | [0.84, 0.89] | [2.77, 4.42] | [0.176, 0.188] | [0.61, 0.74] | [0.85, 0.90] |
| 0.16  | 0.86 | 3.81 | 0.178 | 0.66 | 0.87 |
|       | [0.83, 0.89] | [3.02, 4.65] | [0.171, 0.184] | [0.59, 0.73] | [0.85, 0.90] |
| 0.18  | 0.86 | 3.95 | 0.178 | 0.66 | 0.87 |
|       | [0.84, 0.89] | [3.13, 4.82] | [0.172, 0.184] | [0.59, 0.73] | [0.85, 0.89] |
| 0.20  | 0.85 | 4.75 | 0.176 | 0.62 | 0.85 |
|       | [0.82, 0.87] | [3.84, 5.71] | [0.170, 0.182] | [0.55, 0.69] | [0.83, 0.88] |
| 0.22  | 0.83 | 4.99 | 0.175 | 0.62 | 0.83 |
|       | [0.81, 0.86] | [4.12, 5.88] | [0.168, 0.181] | [0.56, 0.69] | [0.81, 0.86] |
| 0.24  | 0.83 | 4.92 | 0.176 | 0.64 | 0.82 |
|       | [0.81, 0.86] | [4.06, 5.81] | [0.169, 0.182] | [0.58, 0.71] | [0.80, 0.85] |
| 0.26  | 0.83 | 5.41 | 0.174 | 0.64 | 0.80 |
|       | [0.80, 0.85] | [4.53, 6.30] | [0.167, 0.180] | [0.58, 0.70] | [0.77, 0.83] |
| 0.28  | 0.81 | 5.73 | 0.168 | 0.63 | 0.77 |
|       | [0.78, 0.83] | [4.93, 6.58] | [0.161, 0.174] | [0.57, 0.70] | [0.74, 0.80] |
| 0.30  | 0.81 | 5.98 | 0.170 | 0.64 | 0.75 |
|       | [0.79, 0.83] | [5.15, 6.83] | [0.164, 0.176] | [0.58, 0.71] | [0.72, 0.78] |

Table 16: Scramble with tile 32. Results for the metrics for selected noise levels. Shown in square brackets are bootstrapped 95%-confidence intervals. Here topo stands for topographic similarity, conf for conflict count, cont for context independence, pos for positional disentanglement and acc for accuracy.
E Average topographical similarity for random languages

In this section, we compute the average performance of the topographic similarity metric for a random language when a message is of length \( K = 2 \). For simplicity, we assume that the feature space and the alphabet space are the same, and equal \( \mathcal{F} = \{1, \ldots, n\}^2 \). Then the topographic similarity for language \( \ell : \mathcal{F} \to \mathcal{F} \) is defined as

\[
\text{topo}(\ell) = \text{corr}(\rho(\ell(F_0), F_1)), \quad R(\rho(\ell(F_0), \ell(F_1))),
\]

where \( F_0, F_1 \) are uniform random variables on \( \mathcal{F} \) and \( R \) is the rank function. The random variable \( \rho(F_0, F_1) \) takes values \( \{0, 1, 2\} \) with probabilities \( p_0, p_1, p_2 \). Since \( \rho(F_0, F_1) \) is discrete, there are different conventions for defining function \( R \). Typical choices for ranks are: (i) “min-ranks” \( R(0) = 0, R(1) = p_0, R(2) = p_0 + p_1 \); (ii) ”max-ranks” \( R(0) = p_0, R(1) = p_0 + p_1, R(2) = 1 \); (iii) ”average-ranks” \( R(0) = p_0/2, R(1) = (p_0 + p_1)/2, R(2) = (p_0 + p_1 + 1)/2 \). Lemma 1 gives the formula for \( \mathbb{E}_{\ell \sim U}[\text{topo}(\ell)] \).

**Lemma 1.** Let \( R \) be a rank function. Then

\[
\mathbb{E}_{\ell \sim U}[\text{topo}(\ell)] = \frac{p_0 R(0)^2 + \frac{2p_1}{n+1} R(1)^2 + \frac{p_2(n-1)}{n+1} R(2)^2 - \left( \sum_{i=0}^{2} p_i R(i) \right)^2}{\sum_{i=0}^{2} p_i R(i)^2 - \left( \sum_{i=0}^{2} p_i R(i) \right)^2},
\]

where \( U \) is a uniform distribution among all bijective \( \ell : \mathcal{F} \to \mathcal{F} \), \( n_0 = n^2 \), \( n_1 = 2n^2(n-1) \), \( n_2 = n^2(n-1)^2 \), and \( p_i = n_i/n^4 \).

**Proof.** Notice that \( \rho(F_0, F_1) \) and \( \rho(\ell(F_0), \ell(F_1)) \) have the same distribution described by \( p_i \). Define \( \alpha = R(\rho(F_0, F_1)) \) and \( \alpha_{\ell} = R(\rho(\ell(F_0), \ell(F_1))) \). Consequently,

\[
\mathbb{E}_{\ell \sim U}[\text{topo}(\ell)] = \frac{\mathbb{E}_{\ell \sim U}[\alpha] - (\mathbb{E}[\alpha])^2}{\text{Var}(\alpha)}.
\]

Since, \( \mathbb{E}[\alpha] = \sum_{i=0}^{2} p_i R(i) \) and \( \text{Var}(\alpha) = \sum_{i=0}^{2} p_i R(i)^2 - \left( \mathbb{E}[\alpha] \right)^2 \), it remains to compute

\[
\mathbb{E}_{\ell \sim U}[\alpha] = \mathbb{E}[\alpha_{\ell}] = \frac{1}{n^4} \sum_{f_0, f_1 \in \mathcal{F}} R(\rho(f_0, f_1)) \mathbb{E}_{\ell \sim U}[R(\rho(\ell(f_0), \ell(f_1)))]
\]

\[
= \frac{1}{n^4} \sum_{i=0}^{2} \sum_{f_0, f_1 \in \mathcal{F}, \rho(f_0, f_1) = i} R(i) \mathbb{E}_{\ell \sim U}[R(\rho(\ell(f_0), \ell(f_1)))]
\]

\[
= p_0 R(0)^2 + \frac{1}{n^4} \sum_{i=1}^{2} \sum_{f_0, f_1, \rho(f_0, f_1) = i} R(i) \mathbb{E}_{\ell \sim U}[R(\rho(\ell(f_0), \ell(f_1)))]
\]

\[
= p_0 R(0)^2 + \frac{1}{n^4} \sum_{i=1}^{2} \sum_{f_0, f_1, \rho(f_0, f_1) = i} R(i) \left( R(1) \frac{2}{n+1} + R(2) \frac{n-1}{n+1} \right)
\]

\[
= p_0 R(0)^2 + (R(1)p_1 + R(2)p_2) \left( R(1) \frac{2}{n+1} + R(2) \frac{n-1}{n+1} \right).
\]

The second to last equality follows from the fact that there \( n_i(n^2 - 2)! \) bijections \( \ell \) that map \( f_0 \neq f_1 \) to \( \ell(f_0), \ell(f_1) \) such that \( \rho(\ell(f_0), \ell(f_1)) = i \), for \( i = 1, 2 \). \( \square \)

F Optimality of compositional communication

When features are explicitly stated, we can write \( \mathcal{F} = \prod_{i=1}^{K} \mathcal{F}_i \), where \( \mathcal{F}_i \) is the space of values of \( i \)-th feature and \( K \) is the number of features and a message length. We assume that \( |\mathcal{F}_i| = |A_i| \) and that \( |A_i| \geq 2 \). We will assume that a language is a mapping \( \ell : \mathcal{F} \to \mathcal{A}_K \). We said that the language is compositional if a change in one feature only impacts a corresponding index of the message. Formally, we say that \( \ell \) is compositional if and only if for every \( k = 0, \ldots, K \), and \( f_0, f_1 \in \mathcal{F}, \rho(f_0, f_1) = k \iff \rho(\ell(f_0), \ell(f_1)) = k \). Here \( \rho \) stands for the Hamming distance.
Assume Theorem 3.

The following result is a more detailed version of Theorem 2.

The noisy channel transforms a message \( s \) as equation 5). It follows that since \( \rho = \rho(s, s) \) is compositional. Then, there exists \( k > 0 \) such that \( \rho(s, s) = k \).

We start by proving the claim for \( f \in \mathcal{F} \), and denote \( s_0 = \ell(f_0) \). Suppose that \( f \) is not compositional. Then, there exists \( k > 0 \), \( f_1, f_2 \in \mathcal{F} \), \( s_1, s_2 \in A^K \) such that \( \rho(f_0, f_1) = k, \rho(s_0, s_2) = k, \rho(f_0, \ell^{-1}(s_2)) \neq k, \) and \( \rho(s_0, \ell(f_1)) \neq k \). Let \( k^* \) be the biggest among mentioned \( k \)'s and denote \( s_1 = \ell(f_1), f_2 = \ell^{-1}(s_2) \).

By the definition of \( k^* \), we have

\[
\rho(s_0, s_1) < k^*, \quad \rho(f_0, f_2) < k^*.
\]

Since \( \varepsilon < (|A_s| - 1)/|A_s| \), the probability \( \mathbb{P}(s' = s_1|s_0) \) is a decreasing function of \( \rho(s_0, s_2) \) (see equation equation\( \mathcal{F} \)). It follows that

\[
\mathbb{P}(s' = s_1|s_0) > \mathbb{P}(s' = s_2|s_0).
\]
We construct a new language \( \tilde{\ell} \) (see Figure 31).

\[
\tilde{\ell}(f) = \begin{cases} 
    s_1 & f = f_2, \\
    s_2 & f = f_1, \\
    \ell(f) & \text{otherwise.}
\end{cases}
\]

As a result,

\[
\rho \left( f_0, \tilde{\ell}^{-1}(s_1) \right) - \rho \left( f_0, \ell^{-1}(s_1) \right) = \rho \left( f_0, f_2 \right) - \rho \left( f_0, f_1 \right) < 0,
\]

\[
\rho \left( f_0, \tilde{\ell}^{-1}(s_2) \right) - \rho \left( f_0, \ell^{-1}(s_2) \right) = \rho \left( f_0, f_1 \right) - \rho \left( f_0, f_2 \right) > 0.
\]

Putting things together and using the fact that \( H \) is increasing, we get

\[
J_2(f_0, \tilde{\ell}) - J_2(f_0, \ell) = \mathbb{E} \left[ H \left( \rho(\tilde{\ell}^{-1}(\tilde{\ell}(f_0)'), f_0) \right) \right] - \mathbb{E} \left[ H \left( \rho(\ell^{-1}(\ell(f_0)'), f_0) \right) \right]
\]

\[
= \mathbb{P}(s' = s_1|s_0) \left\{ H \left( \rho \left( f_0, \tilde{\ell}^{-1}(s_1) \right) \right) - H \left( \rho \left( f_0, \ell^{-1}(s_1) \right) \right) \right\}
\]

\[
+ \mathbb{P}(s' = s_2|s_0) \left\{ H \left( \rho \left( f_0, \tilde{\ell}^{-1}(s_2) \right) \right) - H \left( \rho \left( f_0, \ell^{-1}(s_2) \right) \right) \right\}
\]

\[
= \{ H \left( \rho(f_0, f_1) \right) - H \left( \rho(f_0, f_2) \right) \} \mathbb{P}(s' = s_2|s_0) - \mathbb{P}(s' = s_1|s_0) < 0.
\]

Consequently, for any non-compositional \( \ell \) we can strictly decrease its loss value \( J_2(f_0, \ell) \). Since the loss is nonnegative and there is a finite number of languages, optimal \( \ell \) has to be compositional. For compositional language \( \ell \),

\[
J_2(f_0, \ell) = \mathbb{E} \left[ H(\rho(s'_0, s_0)) \right] = \sum_{\hat{s} \in A_k^K} H(\rho(\hat{s}, s_0)) \mathbb{P}(s'_0 = \hat{s}|s_0)
\]

\[
= \sum_{k=0}^{K} \sum_{\hat{s} \in A_k^K \rho(\hat{s}, s_0) = k} H(k) \mathbb{P} \left( s'_0 = \hat{s}|s_0 \right) = \sum_{k=0}^{K} H(k) \binom{K}{k} \varepsilon^k (1 - \varepsilon)^{K-k} = \mathbb{E}[H(B_\varepsilon)],
\]

where \( B_\varepsilon \) is a binomial random variable with success probability \( \varepsilon \). This ends the proof for \( J_2 \).

To prove the assertion for \( J_1 \) it is enough to notice that a compositional language satisfies \( \mathbb{P}(f'_j \neq f_j) = \varepsilon \). This follows by equation (6) and the fact that, by symmetry, \( \mathbb{P}(f'_j \neq f_j) \) does not depend on \( j \) (for compositional \( \ell \)).

As a corollary, we see that if \( H(x) = x/K \), \( J_2(f_0, \ell) = \varepsilon \) for any compositional language \( \ell \).

Notice that Theorem 1 follows from the fact that any permutation \( \pi: \mathcal{F} \to \mathcal{F} \) is a bijection.