SHAPING REPRESENTATIONS THROUGH COMMUNICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Good representations facilitate *transfer learning* and *few-shot learning*. Motivated by theories of language and communication that explain why communities with large number of speakers have, on average, simpler languages with more regularity, we cast the representation learning problem in terms of learning to *communicate*. Our starting point sees traditional autoencoders as a single encoder with a fixed decoder partner that must learn to communicate. Generalizing from there, we introduce *community*-based autoencoders in which multiple encoders and decoders collectively learn representations by being randomly paired up on successive training iterations. Our experiments show that increasing community sizes reduce idiosyncrasies in the learned codes, resulting in more invariant representations with increased reusability and structure.

1 INTRODUCTION

The importance of representation learning lies in two dimensions. First and foremost, representation learning is a crucial building block of a neural model being trained to perform well on a particular task, i.e., representation learning that induces the "right" manifold structure can lead to models that generalize better, and even extrapolate. Another property of representation learning, and arguably the most important one, is that it can facilitate *transfer* of knowledge across different tasks , essential for transfer learning and few-shot learning among others (Bengio et al., 2013). With this second point in mind, we can define good representations as the ones that are *reusable*, induce the abstractions that capture the "right" type of invariances and can allow for *generalizing* very quickly to a new task. Significant efforts have been made to learn representations with these properties; one frequently explored direction involves trying to learn *disentangled* representations (Schmidhuber, 1992; Kingma & Welling, 2013; Higgins et al., 2016; van den Oord et al., 2017)), while others focus on general regularization methods (Srivastava et al., 2014; Vincent et al., 2010). In this work, we take a different approach to representation learning, inspired by successful abstraction mechanisms found in nature, to wit *human language* and *communication*.

Human languages and their properties are greatly affected by the size of their linguistic community (Reali et al., 2018; Wray & Grace, 2007; Trudgill, 2011; Lupyan & Dale, 2010). Small linguistic communities of speakers tend to develop more structurally complex languages, while larger communities give rise to simpler languages (Dryer & Haspelmath, 2013). Moreover, we even observe structural simplification as the effective number of speakers grows, as in the example of English language (McWhorter, 2002). A similar relation between number of speakers and linguistic complexity can also be observed during *linguistic communication*. Speakers, aiming at maximizing communication effectiveness, adapt and shape their conceptualizations to account for the needs of their specific partners, a phenomenon often termed in dialogue research as *partner specificity* (Brennan & Hanna, 2009). As such, speakers form *conceptual pacts* with their listeners (Brennan & Clark, 1996), and in some extreme cases, these pacts are so ad-hoc and idiosyncratic that overhearers cannot follow the discussion (Schober & Clark, 1989)!

But how are all these linguistic situations related to representation learning? We start by drawing an analogy between *language* and *representations* induced by the traditional and extensively used framework of autonencoders (AE). In the traditional AE set-up, there is a fixed pair of a single encoder and a single decoder that are trained to maximize a reconstruction loss. However, encoders and decoders co-adapt to one another, yielding *idiosyncratic* representations. The encoders spend representational capacity modeling *any* kind of information about the data that could allow the decoder to successfully reconstruct the input; as long as the encoder and the decoder agree on a representation protocol, this information need not be abstract or systematic. This has a negative impact on the reusability of the representations, something that afterall is a key objective of representation learning. Evidence of this co-adaption is found in the above-mentioned efforts targeting generalization. The human language analogy of the traditional AE setup would be an extreme version of the conceptual pact experiments from Schober & Clark (1989), where two people never communicate with anybody else: the resulting language would be very hard to understand for any outsider.

In this work we test whether removing this co-adaptation between encoders and decoders can yield better generalization, much as dropout removes co-adaptation between activations and thereby yields better generalization in general neural networks. We hypothesize that machines that communicate not with a specific partner but with a multitude of partners, will shape the representations they communicate to be simpler in nature. We introduce a simple framework that we term *community-based autoencoders* (CbAEs), in which there exist multiple encoders and decoders, and at every training iteration one of each is randomly sampled to perform a traditional autoencoder (AE) training step. Given that the identity of the decoder is not revealed to the encoder during the encoding of the input, the induced representation should be such that all decoders can use it to successfully reconstruct the input. A similar argument holds for the decoder, which at reconstruction time does not have access to the identity of the encoder. We conjecture that this process will reduce the level of idiosyncrasy, resulting in representations that are invariant to the diverse encoders and decoders.

We apply CbAEs to two standard computer vision datasets and probe their representations along two axes; their *reusability* and their *structural properties*. We find that in contrast to representations induced within a traditional AE framework 1) the CbAE-induced representations encode abstract information that is more easily extracted and re-used for a different task 2) CbAE representations provide an interface that is easier to learn for new users 3) and the underlying topology of the CbAE representations is more aligned to human perceptual data that are disentangled and structured.

2 Community-based autoencoders

Background One of the simplest and most widely used ways to do representation learning is to train an autoencoder, i.e., encode the input $\mathbf{x}$, usually in a lower-dimensional representation, $\mathbf{z} = e(\mathbf{x}, \theta)$ using some parameters θ, then use the $\mathbf{z}$ representation to decode back the input $\mathbf{x}' = d(\mathbf{z}, \phi)$ through another set of parameters ϕ. θ and ϕ are trained by minimizing a reconstruction loss, e.g.,:

$$L(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'||_2 = ||\mathbf{x} - d(e(\mathbf{x}, \theta), \phi)||_2 \tag{1}$$

The resulting latent vector $\mathbf{z}$ is then treated as the induced representation of the input data, and is often re-used for other problems, such as supervised learning or reinforcement learning. Because this approach is very general and can be applied to any data set, it holds the promise of being able to leverage existing unlabelled data, in order to then quickly solve other problems, using much less data and/or computation. However, the loss in Eq. 1 has an important flaw: it does not directly incentivize the formation of latents that have all the properties of good representations, such as appropriate abstraction and reusability. As a result, significant amounts of research effort have been dedicated to finding a better loss (Vincent et al. (2010), Kingma et al. (2014), *inter alia*).

Our method The CbAE framework (see Figure 1) is inspired by the hypothesis that the size of a linguistic community has a causal effect on the structural properties of its language. Unlike the traditional autoencoder framework, which uses a single encoder paired with a single decoder, the CbAE set-up involves a community of K_{enc} encoders and K_{dec} decoders.[1] As such, we are not dealing with a single autoencoder, but rather a collection of $K_{\text{enc}} \times K_{\text{dec}}$ autoencoders. No single encoder and decoder are associated with one another, but rather the community of encoders are associated with the community of decoders and all combinations may be used together. Importantly, while the network architectures can be (and in fact in this work are) identical across a community (i.e., all encoders and decoders have the same number and organization of units and weights) there is no weight-sharing among members of the community.

[1] For simplicity, in our experiments, we use $K_{\text{enc}} = K_{\text{dec}}$.

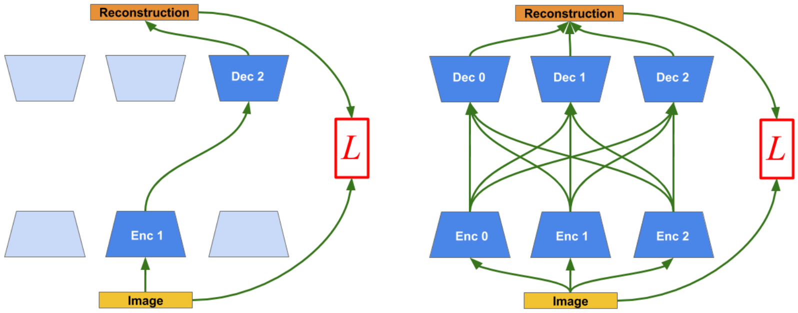

Figure 1: Left: for each iteration, a randomly selected encoder-decoder pair is used. Right: in expectation, all encoders are trained with all decoders, and vice versa.

Training procedure At each training step, given a data point $\mathbf{x}$, we form an autoencoder by randomly sampling an encoder and a decoder from the respective communities. Then, we perform a traditional autoencoding step where we minimize the mean-squared (L_2) loss between the input $\mathbf{x}$ and its decoding (see Eq. 1 and Algorithm 1). Trivially, the traditional autoencoder training protocol can be recovered by setting $K_{\text{enc}} = K_{\text{dec}} = 1$.

Algorithm 1 Community-based autoencoders

```
initialize encoders $\mathcal{E} = \{e_0, ..., e_{K_{\text{enc}}}\}$
initialize decoders $\mathcal{D} = \{e_0, ..., e_{K_{\text{dec}}}\}$
**for** each iteration $i$ **do**
    sample input data $\mathbf{x}_i$
    sample encoder $e_i$ from $\mathcal{E}$
    $\mathbf{z}_i \leftarrow e_i(\mathbf{x}_i)$
    sample decoder $d_i$ from $\mathcal{D}$
    $\mathbf{x}'_i \leftarrow d_i(\mathbf{z}_i)$
    $L_i \leftarrow L(\mathbf{x}'_i, \mathbf{x}_i)$                    ▷ see Eq. 1
    optimize $e_i$ and $d_i$ with respect to $L_i$
**end for**
```

There are two main reasons why we think this will have a positive effect on the quality of the representations. First, given that the chosen encoder e_i for iteration i does not have *a priori* information about the identity of the chosen decoder d_i, and given that there are a number of decoders all with different weights, the encoder should produce a latent z_i that is potentially decodable by all different decoders. Similarly, given that each decoder d_i receives over its training lifetime latents from a number of different encoders, the decoder should learn to decode representations produced by all encoders. We hypothesize that this training regime will produce latents that are less prone to have idiosyncrasies rooted in the co-adaptation between a particular pair of encoder and decoder.

Relation to dropout The CbAE setup is reminiscent of *dropout* (Srivastava et al., 2014): The entire community can be viewed as one much larger and highly parallel model, from which at each iteration a selection of weights (corresponding to one specific community member) is chosen. However, a crucial difference is that here, the choice of weights happens in a very correlated way; it is not a random set of weights, but one of K_{enc} or K_{dec} non-overlapping subsets that is selected at each training step. As a consequence, the weights in one community member (e.g. an encoder) will be much slower to adjust (if at all) to those in the rest of the community, and a higher degree of diversity is maintained. There is some mutual adjustment, of course, but it is a second-order effect: encoder e_i and encoder e_j will only get information about each other's encoding through the gradients of decoders that have learned to decode them.

The curse of co-adaptation The goal of our method is to avoid co-adaptation between the encoder and decoder. However, due to their flexibility, neural networks are in principle capable of co-adapting to several partner modules at once. As a consequence, the encoders can avoid convergence and still learn to produce latents from which the decoders can successfully reconstruct the input by capitalizing on encoder-specific information. Intuitively, we can think of this as the encoder essentially "signing" the latents with their unique ID. We test whether this indeed manifests in the setup by training a linear classifier whose task is to identify the encoder from the latent representation: $p_e(\mathbf{z}) = \exp(\mathbf{w}_e^T \mathbf{z}) / \sum_{e'} \exp(\mathbf{w}_{e'}^T \mathbf{z})$.

As Table 1 shows, the encoder classifiers perform significantly better than chance in spite of having to keep up with shifting representations, indicating that pairwise co-adaptation does indeed happen to some extent for all community sizes. The non-monotonous behaviour seen on the row labelled *without entropy loss* is due to the two competing effects. As the community size grows, the encoder identification task becomes harder (hence the lower chance), and the error rate naturally increases. However, larger communities also lead to slower rates of representation shift for every individual encoder, making it easier for the encoder classifier to keep up with their changing representation.

	Community size			
	2	4	8	16
Chance	*0.5*	*0.75*	*0.875*	*0.938*
No entropy loss	0.3	0.48	0.47	0.28
With entropy loss	0.52	0.764	0.875	0.937

Table 1: Encoder identification error rate on MNIST.

A similar phenomenon of co-adaptation is often encountered in domain-adaption neural frameworks. To alleviate this, adversarial losses or gradient reversal layers (Ganin et al., 2016) are introduced to penalize representations from retaining domain-specific information. Here, in order to counteract the all-to-all pairwise co-adaptation effect, we add a simple adversarial loss forcing the encoders to be indistinguishable for the encoder classifier while keeping the encoder classifier itself fixed. In particular, the extra loss term is the negative entropy of the classifier, $L_{\text{entropy}}(\mathbf{z}) = \sum_e p_e(\mathbf{z}) \log p_e(\mathbf{z})$.

Training of CbAE We use MNIST and CIFAR-100, with community sizes of 1, 2, 4, 8 and 16. The batch size is fixed at 128 throughout all experiments. We use the Adam optimizer with a learning rate of 10^{-4}. The encoders are straightforward convolutional neural networks of VGG-flavour, with depths of 6 (MNIST) and 10 (CIFAR-100) layers respectively. For the details we refer the reader to the Appendix. The decoders implement the corresponding transpose convolutions.

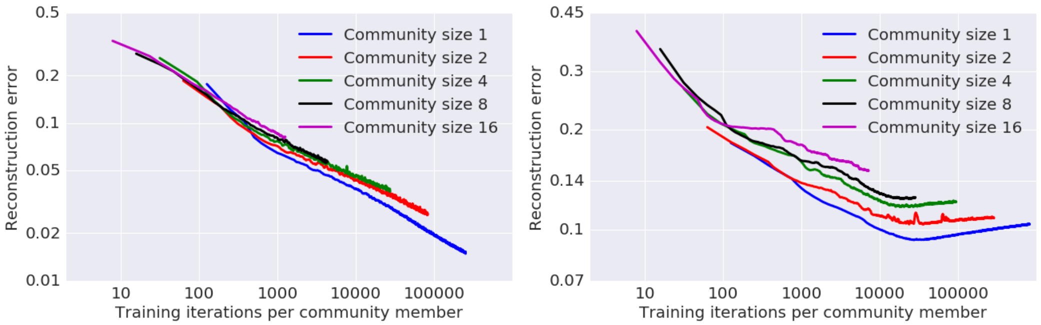

Figure 2: The reconstruction error per pixel on MNIST (left) and CIFAR-100 (right).

Having to respond to more communication partners makes the job of the individual encoders and decoders harder. This effect is seen in the reconstruction loss (see Figure 2, where an increase in community size leads to a penalty in the reconstruction error even when correcting for the amount of training data seen by each community member. Note that this is not necessarily limiting when considering the desired properties of the representation, since the pixel-loss is merely a *self-supervision* signal: some pixel-level information is lost, but ultimately pixel-level information is not the true goal of the representation learning exercise, as discussed in Section 1. The interesting question is however: given the capacity of the latents, are we trading-off reconstruction performance for other more relevant properties such us reusability or structure? The experiments presented in section 3 aim at answering precisely this question.

3 Assessing the Quality of Representations

In the previous section we introduced the CbAE set-up and found that the reconstruction loss increases as the community size grows. However, the reconstruction loss in this setup is just a learning signal for representation learning, rather than the end goal. Ultimately, we are interested in good representations that could allow for generalization, knowledge transfer and reusability. In this section, having trained the CbAEs, we devise a number of parametric (Section 3.1) and non-parametric (Section 3.2) evaluation methods that probe the representations for exactly these properties.

3.1 Parametric probe tasks

3.1.1 Tasks and metric

Training new encoders and decoders Human languages have the property that the more regular and systematic they are, the easier they are for learners to acquire. We examine whether the latent interface found by the CbAE setup has the same property, i.e., is easier to learn for new users. To do so, we train newly initialized encoders and decoders. The hypothesis is that if the CbAE-trained encoders and decoders have learned to encode information in the representation in a systematic way, rather than in an ad-hoc and idiosyncratic way, this would result in the new, untrained, encoders and decoders being able to learn the representation with less *effort*, which we define operationally as better sample complexity (see below). This evaluation task is illustrated in the two leftmost panels of Figure 3.

Transferring representations to a new task Next, we investigate the transfer capabilities of the representations to a different task; we freeze the CbAE encoders, perform supervised learning on image classification by training *linear* classifiers and evaluate their sample complexity. The hypothesis is that the CbAE framework induces abstract representations of the input data that would allow a simple linear classifier to achieve a better sample complexity.

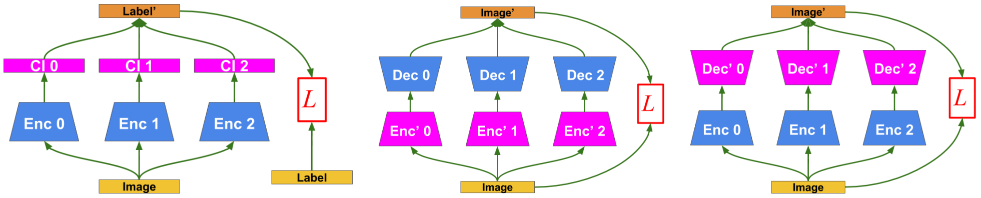

Figure 3: Set-up for assessing the quality of the representations. Purple boxes represent trainable evaluation modules; blue boxes represent trained encoders and decoders that are kept fixed during the evaluation training. Left: linear classifiers; centre: new encoders; right: new decoders.

Experimental setup In these probe tasks, only the newly initialized evaluation modules (new encoders, new decoders, and linear classifiers) are trained, and the encoders and decoders trained in the CbAE setup are frozen. The tasks share the same basic set-up: all frozen members of the community of encoders or decoders are coupled individually to an evaluation module, which is trained to perform best-response to their pre-trained partner.

For the new encoders and decoders, we use the same architecture as for the CbAE-trained ones. We use the Adam optimizer with a learning rate of 10^{-4}. For the linear classifiers, we fit a linear layer, followed by a softmax, on the latents of each CbAE-trained encoder. We use the Adam optimizer with a learning rate of 10^{-3} and optimize the cross-entropy between the predicted label $\hat{y}$ and the actual label y. We use a minibatch size of 128 throughout all experiments.

Sample complexity gain For every parametric probe task, we first record the average performance achieved by all modules trained with a given community after a given number of training iterations. We then obtain the number of training iterations needed for the traditional AE (i.e., community size 1) to reach the same performance, and compute the ratio of these two training durations as the *sample complexity gain*.

The above formulation takes the following form in more mathematical notation: Given a learning curve $L(i)$ which maps an iteration i to an obtained result $L(i)$, we define the *inverse learning curve*:

$$L^{\text{inv}}(L') = \min_{\text{s.t.} L(i) \leq L'} i \tag{2}$$

The inverse learning curve returns the first iteration at which the result dropped below the argument L'. Equipped with this function, the sample complexity gain of curve L at iteration i relative to curve L_{baseline} is straightforward to compute:

$$\text{SCG}(L, i, L_{\text{baseline}}) = \frac{i}{L^{\text{inv}}_{\text{baseline}}(L(i))} - 1 \tag{3}$$

If the two curves L and L_{baseline} are identical, $L^{\text{inv}}_{\text{baseline}}(L(i)) = i$, and $\text{SCG}(L, i, L_{\text{baseline}}) = 0$ as expected. A negative sample complexity gain indicates that L reaches the value $L(i)$ at a later iteration than L_{baseline}, i.e. $L^{\text{inv}}_{\text{baseline}}(L(i)) < i$.

3.1.2 Parametric probe results

Transferring representations to a new task After training the image classifier, we evaluate its performance on the test set. Figure 4 shows the sample complexity gain relative to the traditional AE case Overall, we find that classifiers trained on the latents learned by larger communities learn faster. For example, the leftmost bar on the MNIST plot shows that a classifier trained on the latents from community of size 2 needs 8 iterations to reach a performance that takes almost twice (1.6 times) as many iterations for a classifier trained on the latents from a classical AE. Moreover, the MNIST plot clearly shows that larger communities lead to faster classifier learning. As the classifier training progresses, the gains relative to the classical AE become smaller; this suggests that there is might still be some co-adaption, which however is significantly delayed by the introduction of the community. This positive effect of the community is also clear since the largest community is still speeding up relative to the classical AE after 32 iterations. In the case of CIFAR-100, we observe similar sample complexity gains, but the community size effect appears to be reversed. We attribute this to the larger model used (and needed) for this data set, which presents a significantly more complex task both for autoencoders and for classifiers. This larger model in turn requires more CbAE iterations for the communities to learn to represent the data at all, with each community member only seeing $1/K$ of the CbAE iterations (K being the community size).

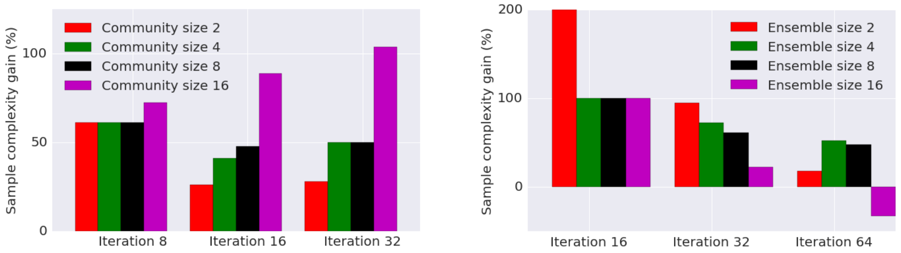

Figure 4: The sample complexity gain relative to the traditional AE setup when training a linear classifier, on MNIST (left) and CIFAR-100 (right).

Training new encoders and decoders Figure 5 shows the results for training new decoders on MNIST and CIFAR-100. The new decoders learn faster with encoders trained in larger communities. Moreover, we observe that although there are large sample complexity gains over the baseline, in absolute numbers these gains are smaller than the ones obtained in the previous transfer task. While the image classification task requires a CbAE encoder to have produced a representation capturing a certain level of abstraction, training a new decoder requires the CbAE encoder to represent precise information about the data. The fact that CbAEs are better in the former than the latter suggests that their representations are more abstract in nature. Evidently, we are ready to accept this trade-off; as discussed in the introduction, abstraction is the holy grail of representation learning.

Figure 6 shows the results for training new encoders on MNIST and CIFAR-100. The new encoders learn faster with decoders trained in larger communities, and display roughly the same sample complexity gain pattern as the new decoders, albeit with slightly better absolute numbers. To understand

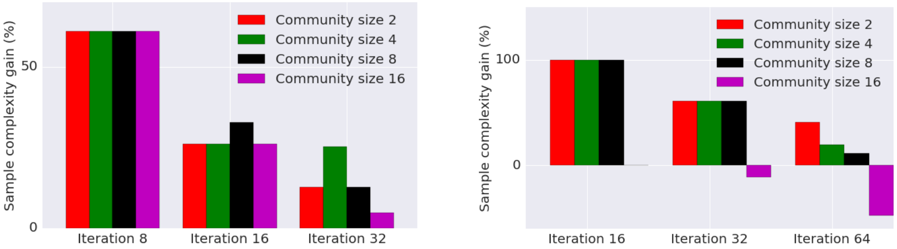

Figure 5: The sample complexity gain relative to the traditional AE setup when training new decoders, on MNIST (left) and CIFAR-100 (right), while training new encoders.

this, we note that there is an asymmetry between encoders and decoders, in that the decoders can learn to decode a large hypervolume in latent space into roughly the same image, encompassing the encodings of all individual encoders. A new encoder then only has to learn to encode an image somewhere into that hypervolume to get a reasonable reconstruction error. A new decoder, however, has to learn best-response to a specific encoder, which essentially involves more adaptation to residual idiosyncrasies and can therefore be a more difficult task.

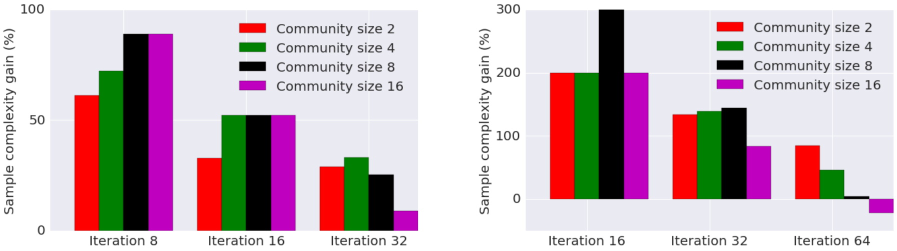

Figure 6: The sample complexity gain relative to the traditional AE setup when training new encoders, on MNIST (left) and CIFAR-100 (right), while training new encoders.

Comparison to other regularization mechanisms We have performed the same probe analyses presented in sections 3.1.1 and 3.1.2 on the representations learned by a traditional AE setup enhanced with (neuron-level) dropout, and found no gains relative to the dropout-free setting. Exploring variations on dropout that interpolate between large expected overlaps between subsets (the traditional implementation) and fully mutually exclusive subsets (our method) is an interesting direction of future work. Moreover, all our models are trained with batch normalization, indicating that the gains we find are orthogonal to the specific regularization advantages it provides.

3.2 Non-parametric probe task: Latent space structural analysis

Finally, we ask the question of to what degree the speaker-invariance bias imposed by the CbAE framework induces abstract representations that share the same underlying *structure* with human perceptual data.[2] As a proxy of human perceptual data, we use the Visual Attributes for Concepts Dataset (VisA) of Silberer et al. (2013), which contains human-generated per-concept attribute annotations for concrete concepts (e.g., `cat`, `chair`, `cat`) spanning across different categories (e.g., `mammals`, `furniture`, `vehicles`), annotated with general visual attributes (e.g., `has_whiskers`, `has_seat`). Table 2 presents some examples of the conceptual representations found in VisA. As we can see, concepts representations are *structured* and *disentangled*. Therefore, achieving high similarity with these would indicate that the CbAE-induced representations encode similar conceptual abstract information. Most importantly, this is an independent task and requires *no* additional training of parameters.

[2] We note that this experiment is only conducted for CIFAR-100 as this dataset contains categories for common nouns (e.g., cat, dog, chair) for which we can meaningfully probe humans for perceptual similarity (as opposed to the numerical categories found in MNIST).

Representation Similarity Analysis For measuring the similarities between the human perceptual data and the CbAE-induced representations, we perform Representational Similarity Analysis (RSA) in the two topologies, a method popular in neuroscience (Kriegeskorte et al., 2008). For each community configuration, we sample 5,000 images and encode them with all encoders. Following that, for each encoder-specific set of latents, we apply concept-based late fusion, meaning that we average in a single latent all latents belonging to the same concept, to arrive to 68 concept-based representations. We then compute two sets of pairwise similarities of the 68 concepts, i.e., one set using their concept-based CbAE-induced latent representations and one set the concept-based VisA attribute representations. With these two lists of cosine similarities in hand, the RSA between the two topologies is taken as the Spearman correlation of these two lists of similarities. Given that RSA is a second-order similarity, we are not asking the question of how similar (say in terms of cosine) the two spaces are, but rather how similar their topology is, i.e., whether points that are nearby in the latent space are also nearby in the VisA space.

Results Table 3 summarizes our results. For each community configuration we report the mean RSA performance (obtained by averaging the RSA scores produced by the different encoders) and the maximum performance. To account for the potential confounder caused by the different initializations in the CbAE, we compare the results with the best result found from same number of independent AE. We observe that the mean similarity increases with the size of the population, confirming the hypothesis that CbAE produce on average abstract representations that to some degree reflect the topology of the highly structured and disentangled human data.

Moreover, we observe even higher gains when looking at the best RSA value within a CbAE; training an encoder within a community of diverse partners can lead to more abstract and structured representations than training a diverse set of independent encoders each with a fixed decoder partner. This result rejects an alternative hypothesis; the gains cannot by explained just by increasing the diversity of the initializations, it is the community training of these diverse models that leads to increases performance.

Finally, while the largest CbAE (i.e., community size 16) has higher RSA similarity than the baseline, it shows the smallest gains, a pattern consistent with the the rest of the parametric probe results of CIFAR-100. We attribute this to the fact that this community had the smallest number of iterations per member, and had therefore not had the opportunity to learn to represent the data well yet.

	cat	chair	car
has_whiskers	1	0	0
has_seat	0	1	1
made_of_metal	0	0	1
has_legs	1	1	0

Table 2: Examples of conceptual representations in the VisA dataset.

Community size	mean ρ	max ρ
1	0.341	0.314
2	0.355	0.382
4	0.372	0.389
8	0.401	0.423
16	0.352	0.369

Table 3: Perceptual similarity between CbAE-induced and VisA representations.

4 Discussion

We have presented Community-based AutoEncoders, a framework in which multiple encoders and decoders collectively learn representations by being randomly paired up on successive training iterations, encouraging a similar lack of co-adaptation that dropout does at the activation level, at model level. Analogous to the structural simplicity found in languages with many speakers, we find that the latent representations induced in this scheme are *easier to use* and more *structured*. This result is philosophically interesting in that it suggests that the community size effects found in human languages are general properties of any representation learning system, opening avenues to potential synergies between representation learning linguistics.

The price for obtaining these representations is the increase in computational requirements, which is linear in the community size. Due to the reusability of the resulting representations, this cost may be amortized over a number of applications trained on top of the encoders. Furthermore, the community-based training procedure is highly parallelizable, since only the latents and corresponding backpropagated errors need to be sent between the encoders and decoders.

APPENDIX

CNN ARCHITECTURES USED

Kernel size	Stride	Channels
3x3	2	64
3x3	2	64
3x3	2	128
3x3	2	128
3x3	2	256
3x3	2	256

Table 4: The CNN architecture used in the MNIST experiments.

Kernel size	Stride	Channels
3x3	1	32
5x5	2	64
3x3	1	64
5x5	2	128
3x3	1	128
5x5	2	256
3x3	1	256
5x5	2	512
3x3	1	512
2x2	2	512

Table 5: The CNN architecture used in the CIFAR-100 experiments.

REFERENCES

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Susan E Brennan and Herbert H Clark. Conceptual pacts and lexical choice in conversation. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 22(6):1482, 1996.

Susan E Brennan and Joy E Hanna. Partner-specific adaptation in dialog. *Topics in Cognitive Science*, 1(2):274–291, 2009.

Matthew S. Dryer and Martin Haspelmath (eds.). *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig, 2013. URL `https://wals.info/`.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Irina Higgins, Loic Matthey, Xavier Glorot, Arka Pal, Benigno Uria, Charles Blundell, Shakir Mohamed, and Alexander Lerchner. Early visual concept learning with unsupervised deep learning. *arXiv preprint arXiv:1606.05579*, 2016.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pp. 3581–3589, 2014.

Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2:4, 2008.

Gary Lupyan and Rick Dale. Language structure is partly determined by social structure. *PloS one*, 5(1):e8559, 2010.

John McWhorter. What happened to english? *Diachronica*, 19(2):217–272, 2002.

Florencia Reali, Nick Chater, and Morten H Christiansen. Simpler grammar, larger vocabulary: how population size affects language. *Proc. R. Soc. B*, 285(1871):20172586, 2018.

Jürgen Schmidhuber. Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–879, 1992.

Michael F Schober and Herbert H Clark. Understanding by addressees and overhearers. *Cognitive psychology*, 21(2):211–232, 1989.

Carina Silberer, Vittorio Ferrari, and Mirella Lapata. Models of semantic representation with visual attributes. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 572–582, 2013.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15 (1):1929–1958, January 2014. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2627435.2670313.

Peter Trudgill. *Sociolinguistic typology: Social determinants of linguistic complexity*. Oxford University Press, 2011.

Aaron van den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pp. 6306–6315, 2017.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010.

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