Disentangled latent representations of images with atomic autoencoders

Alasdair Newson[1] and Yann Traonmilin[2]
[1]Telecom Paris, Paris, France. alasdair.newson@telecom-paris.fr
[2]CNRS,Univ. Bordeaux, Bordeaux INP, IMB, UMR 5251,F-33400 Talence, France.
yann.traonmilin@math.u-bordeaux.fr

***Abstract*—We present the atomic autoencoder architecture, which decomposes an image as the sum of elementary parts that are parametrized by simple separate blocks of latent codes. We show that this simple architecture is induced by the definition of a general atomic low-dimensional model of the considered data. We also highlight the fact that the atomic autoencoder achieves disentangled low-dimensional representations under minimal hypotheses. Experiments show that their implementation with deep neural networks is successful at learning disentangled representations on two different examples: images constructed with simple parametric curves and images of filtered off-the-grid spikes.**

I. Introduction

The autoencoder is a well-known neural network architecture whose goal is to project data to and from a *latent space*, which is usually of much smaller dimensionality than the original data space, this feature being useful for a wide variety of tasks. Given a collection of samples $X = (x_1, \dots, x_N)$ with $x_i \in \mathbb{R}^n$, autoencoders provide a low dimensional representation of the x_i by using an encoder/decoder pair: the autoencoder f is the composition of an encoder f_E and a decoder f_D and is typically trained by minimizing the quadratic reconstruction loss $\mathcal{L}_X(f) := \sum_{i=1}^N \|f(x_i) - x_i\|_2^2$, i.e.

$$f^* = f_D^* \circ f_E^* \in \arg\min_{f\in\mathcal{F}} \mathcal{L}_X(f) \tag{1}$$

where $\mathcal{F}$ is the set of autoencoders having a given architecture. Typically, the encoder f_E projects data samples to a low-dimensional "latent" space $\mathbb{R}^d$, and the decoder f_D performs the opposite operation, producing data in $\mathbb{R}^n$ from a latent code. In practice, estimating f^* as a deep neural network parametrized by its weights and biases has shown a huge number of applications from solving inverse problems [17] to photo realistic image rendering which employs this architecture in recent diffusion models.

Autoencoders provide a *generative* model of the considered data, i.e. data points can be generated as the image of low dimensional points by the decoder. However, we can not only generate but also edit or manipulate data by moving in the latent space. Towards this goal, several fundamental questions arise: how can one navigate the latent space with the guarantee that the chosen path does not leave the latent set? Is it possible to manipulate latent codes in a meaningful way? Such questions are linked to what is known as the "disentanglement" problem. Indeed, the ultimate goal of generative models is to establish a latent space where each coordinate corresponds to a given feature of the data, thus disentangling these features. For example, for facial images, this could be high-level attributes such as an expression. Thus, editing an image would correspond simply to moving along a coordinate in the latent space. In this article, we focus on low-level features of signal and images such as positions, amplitudes and shapes of objects.

Moreover, we remark that autoencoders are heavily linked to the theory of low-dimensional representation of data that has seen much development in the past twenty years with the theory of sparse recovery and compressed sensing and its extensions (see [9] for an overview). Thus, it appears natural to consider the disentanglement problem from the viewpoint of this theory.

a) Contributions: In this paper, we design an autoencoder architecture tailored for the learning of sparse atomic models. Examples of such models include sketch images and off-the-grid sums of spikes (used in microscopy, astronomy, echo retrieval etc.). This design leads to disentangled low-lvel low-dimensional representations that can be manipulated in a meaningful way.

We define in Section II the *atomic autoencoder*. We show how this architecture is induced by the definition of the general atomic model of the data and discuss its practical implementation with deep neural networks (DNN). We show two properties of such autoencoders: under a perfect learning hypothesis, they provide a naturally disentangled representation (for a specific notion of *atomic disantanglement* defined in this Section). Secondly, considered atoms should be indivisible to avoid entanglement. Note that these properties are elementary from a mathematical perspective and are mainly induced by the atomic structure.

In Section III, we use atomic autoencoders for the decomposition of images composed of parametric curves and for the estimation of off-the-grid spikes. In both cases, we access a disentangled representation and manipulate this representation in a meaningful way: we modify the local shape of the image in the first and access the position and amplitudes of the spikes in the second example. This shows that we have access to a disentangled latent representation with *no more constraints* in the training than the general architecture of the atomic autoencoder. To the best of our knowledge, no other DNN architecture provides the same features.

Code You can find and use the code corresponding to this work at https://github.com/alasdairnewson/atomic_ae/.

b) Related work: The goal of automatically discovering a hidden parametrisation of a class of objects is the objective of representation learning. Autoencoders are an ideal setting for this problem, and have existed for a long time, starting in the 1980's [1], [4], [8]. The work of Cheung et al. [7], Kumar et al. [13] and Lezama [16] attempt to achieve disentanglement in autoencoders by incorporating a covariance loss function into the training process. Lample et al. [14] proposed Fader networks, which try to isolate a single image characteristic in a single latent component, with an innovative use of a discriminator network. This produces a network where the characteristic can be effectively controlled with a slider. Finally, the authors of β-VAE B [10], β-VAE H [5], FactorVAE [12] and β-TCVAE [6], propose frameworks or regularisation to disentangle variational autoencoders by weighting the Kullback-Leibler divergence term to encourage factorised representations in the latent space.

Our approach is closely related to dictionary learning (see e.g. [19] for an overview), where signals of interest are decomposed on a family of vectors (atoms). For example, early works on nonnegative matrix factorization for image decomposition show how a dictionary with sparse features can be used to represent an image [15]. Atomic autoencoders can be seen as a continuous version of dictionary learning. Extension to continuous dictionary with DNN has been proposed using composition of autoencoders in [18]. However, this architecture does not permit to isolate simple features in a given block of latent code like atomic autoencoders.

Finally, recent work [20] indicates that trained DNNs can learn the simplest model, i.e. this model can be learned with minimal distortion, even if worst case bounds suggest that it is difficult to achieve in practice [3]. We note that the general architecture proposed in this paper shares some similarities with "branching" [11], however our architecture is geared towards disentangled latent representations, which is novel.

II. Atomic autoencoders: from sparse models to disentangled representations with DNN

In this section, we introduce the *atomic autoencoder* architecture as a consequence of considering a class of generalized sparse models, we discuss some of its elementary properties and their implementation with DNNs.

A now almost canonical sparsity model is the following:

$$\Sigma_{\mathcal{A},k} = \{x : x = \sum_{i=1}^{k} a_i, a_i \in \mathcal{A}\} \tag{2}$$

where $\mathcal{A}$ is a set of *atoms* (often called a dictionnary) and k is the number of atoms required to represent the data x. For example, in classical sparsity, atoms are weighted vectors of sparsity 1. Such models are ubiquitous in signal and image processing and have been proven powerful for the resolution of ill-posed inverse problems.

In the following, we consider first an ideal data model and its induced *ideal* atomic autoencoder. We make the hypothesis

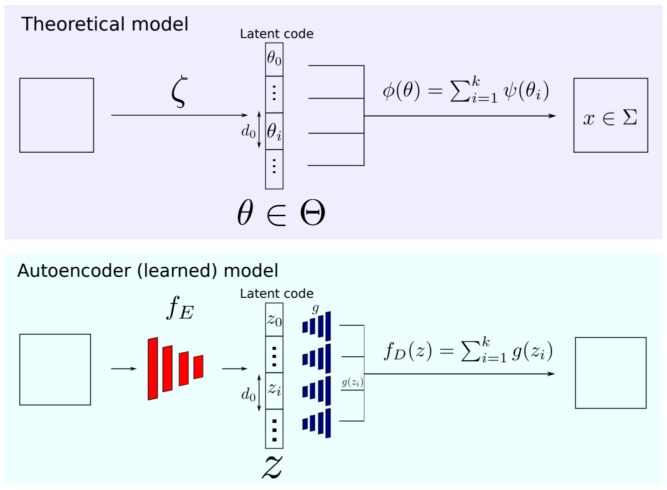

Fig. 1. **Atomic autoencoder model.** In this model, we consider a ideal latent space Θ. Starting from an ideal latent code $\theta \in \Theta$, an ideal data point is generated by the sum of *the same* function ψ applied to k sub-blocks of θ: $\phi(\theta) = \sum_{i=1}^{k} \psi(\theta_i)$, thus defining an *atomic decoder* architecture.

that the x_i belong to a model defined by Equation (2). Of course, a more realistic modeling would consider x_i which approximately belong to $\Sigma_{\mathcal{A},k}$ (e.g. by considering that some distance $d(x_i, \Sigma_{\mathcal{A},k})$ between the x_i and $\Sigma_{\mathcal{A},k}$ is bounded), however the impact of this approximation goes beyond the scope of this paper and is left for future work. Afterwards, we consider a *learned* neural network architecture which approximates this ideal model. Recent work indicates [20] that given a large enough dataset the ideal model can be learned up to a given distortion.

A. Ideal atomic autoencoders: definition and elementary properties

We suppose that elements $x \in \Sigma_{\mathcal{A},k} \subset \mathbb{R}^n$ are generated from a vector of parameters θ, via a function ϕ, i.e. $x = \phi(\theta)$. We refer to θ as the *ideal latent code*.

In the literature of sparse representations, it is often supposed that the set of atoms $\mathcal{A}$ has some additional structure. It can be parametrized using a function $\psi : \mathbb{R}^{d_0} \to \mathbb{R}^n$. Hence, each atom can be written $a_i = \psi(\theta_i)$ and for any $x \in \Sigma_{\mathcal{A},k}$,

$$x = \phi(\theta) = \sum_{i=1}^{k} \psi(\theta_i) \tag{3}$$

where $\theta_i \in \mathbb{R}^{d_0}$ represents a sub-block of size d_0 of the ideal latent code[1]. We refer to a coordinate of a sub-block θ_i as a latent *coordinate*, and note $\theta_{i,j}$ the j^{th} coordinate of the i^{th} block. We have $\theta = (\theta_1, \ldots, \theta_k) \in \mathbb{R}^{kd_0}$, as in Equation (3). We refer to ϕ as the *ideal decoder*: data points $x \in \Sigma_{\mathcal{A},k}$ can be coded with kd_0 parameters, using ϕ (and ψ). The ideal atomic data model as illustrated in Figure 1 is given by:

[1]We also refer to a sub-block θ_i simply as a latent block

$$\Sigma_{\mathcal{A},k} = \Sigma_{\psi,k} := \{x : x = \sum_{i=1}^{k} \psi(\theta_i), \theta_i \in \Theta_0\}, \tag{4}$$

where $\Theta_0 \subset \mathbb{R}^{d_0}$ is the set in which individual blocks of ideal latent codes live. We suppose that the ideal latent set is $\Theta = \phi^{-1}(\Sigma_{\psi,k}) = \Theta_0^k \subset \mathbb{R}^{kd_0}$ (we start out with the ideal data, and find the ideal latent set via ϕ^{-1}). Note that low-dimensional representations, i.e. $kd_0 << n$, are often sought after; e.g. to serve as a prior model in inverse imaging problems.

As an example, for off-the-grid spikes convolved with a Gaussian kernel, we can set $\psi(\theta) = \psi(a,t) = a_i G(t)$ where $\theta = (a,t)$, G is a Gaussian function centered at t, and a_i is the amplitude of the spike. Hence the function ψ parametrizes individual spikes with their amplitude and position. In another case, with images of non-overlapping disks, ψ is a function which produces an image of a disk from the position and size of the disk.

From the definition of $\Sigma_{\psi,k}$, we see that there exists $\zeta : \mathbb{R}^n \to \Theta$, such that $\phi \circ \zeta(x) = x$ for any $x \in \Sigma$: just define ζ that arbitrarily choses one of the representations of x in Θ (e.g. using lexicographical order). Indeed, due to the sum in Equation (3), the ordering of the latent blocks does not matter. We call ζ the *ideal atomic encoder*. We have just defined the *ideal atomic autoencoder* $\phi \circ \zeta$ induced by the model with the following constraints: the latent set has a block structure, ie $\theta = (\theta_1, \dots, \theta_k) \in \mathbb{R}^{kd_0}$, and the decoder is the sum of the same function ψ of different blocks of latent codes.

To provide useful disentangled representations, the main ingredient is to perform encoding with the smallest dimension possible. We formalize this by supposing that the ideal latent set is in a (smooth) bijection with the cube $[0,1]^{kd_0}$ (i.e. it is a smooth manifold). In other words, there is no space left around Θ for codes that do not produce an element of Σ. This relies on the fact that there is no smooth bijection from $[0,1]^{d'}$ to $[0,1]^d$ if $d' < d$. Given sets U, V, let $\mathcal{C}_1(U,V)$ be the set of continuously differentiable functions from U to V (networks are continuously differentiable almost everywhere, activations can also be smoothed if necessary).

Assumption II.1 (Filling latent set)**.** *We say an atomic autoencoder $\phi \circ \zeta$ of Σ yields a filling latent set $\Theta = \phi^{-1}(\Sigma)$ if there is a bijection $h \in \mathcal{C}^1([0,1]^{kd_0}, \Theta)$ with $\mathcal{C}^1$ inverse between $[0,1]^{kd_0}$ and Θ.*

Now that we have defined the atomic autoencoder, we come back to our initial question: disentanglement. An ideal atomic autoencoder that verifies Assumption II.1 has the two following properties. Firstly, as Θ is in a smooth bijection with $[0,1]^{kd_0}$, given a point θ in the interior of Θ there is an open set containing θ such that all points of this set parametrize an element of Σ: we can generate elements of the model set Σ freely and navigate safely in all directions of the latent set, without "falling outside", which is a direct consequence of Assumption II.1. Of course we would need to know the function h to do this fully, without this knowledge we can still do it locally. Secondly, thanks to the intrinsic atomic structure, we can modify individually each code to change only one "simple" feature of x at a same time. If $\theta \in \Theta$ and we want to modify one latent block i by a (sufficiently) small change Δ_i, the previous property ensures that $\theta + (0, \dots, 0, \Delta_i, 0, \dots, 0) \in \Theta$.

We call the combination of these two properties *atomic disentanglement*. Given a low-dimensional model and an autoencoder, it is verified if Assumption II.1 is verified.

Lemma II.1. *Suppose $\Sigma, \Theta, \phi \circ \zeta, h$ verify Assumption II.1 and $\mathrm{int}(h^{-1}(\Theta)) \neq \emptyset$ (where int denotes the interior). Let $\theta \in \Theta$ such that $h^{-1}(\theta) \in \mathrm{int}(h^{-1}(\Theta))$. Then there exists an open set O of $\mathbb{R}^{kd_0}$ such that $\theta + O \subset \Theta$.*

An example of an ideal autoencoder that achieves atomic disentanglement is one which addresses the off-the-grid sparse spike estimation problem. Indeed, in this case, we have $\Sigma = \left\{ aG(t_1) + bG(t_2), a, b \in [a_{min}, a_{max}]; t_1, t_2 \in [0,1]^d, t_1 \neq t_2; \right\}$. Note that in many cases, if Σ contains an element composed of k atoms, then any "simpler" signal composed of less atoms is still in the model (as is usually done in classical sparsity model), i.e. for all $(\theta_i)_{i \in I}$, with $I \subset \{1, \dots, k\}$, we have that $\sum_{i \in I} \psi(\theta_i) \in \Sigma$. We observe in experiments that trained atomic DNN autoencoders seem to have this property without any explicit constraint for this in the training.

We can also show two important facts (see supplementary material). Firstly, perfectly trained atomic autoencoders achieve atomic disantanglement. Secondly, it must not be possible to split the function ψ into simpler functions if we want to ensure that a latent code encoding a given feature is not mixed across latent blocks. This is natural, since we want the function ψ to decode fundamental features of the image (penstrokes, spikes etc.).

B. A tailored neural architecture

We showed that a large class of datasets can be represented with atomic auto-encoders. In the experimental part of this article, we train autoencoders using (leaky) ReLu DNN architectures. It is natural to train autoencoders $f_D \circ f_E$ with the structure $f_D(z) = \sum_{i=1}^{k} g(z_i)$, i.e. the decoder is made of k identical blocks which approximate (up to a bijection) $\phi(z) = \sum_{i=1,k} \psi(z_i)$ and the encoder must approximate ζ. We highlight the fact that the decoder block g is indeed repeated, so that during training, its parameters are the same when applied to all z_i's (see Figure 1 for a representation of the atomic autoencoder).

We highlight the fact that the decoder block g is indeed duplicated, so that during training, its parameters are the same when applied to all z_i's. Note that the symmetrical architecture proposed by early works on autoencoders does not have any really satisfying theoretical explanation. It is known from low-dimensional recovery that low-dimensional signals can be encoded almost universally by *linear* encoders while decoders

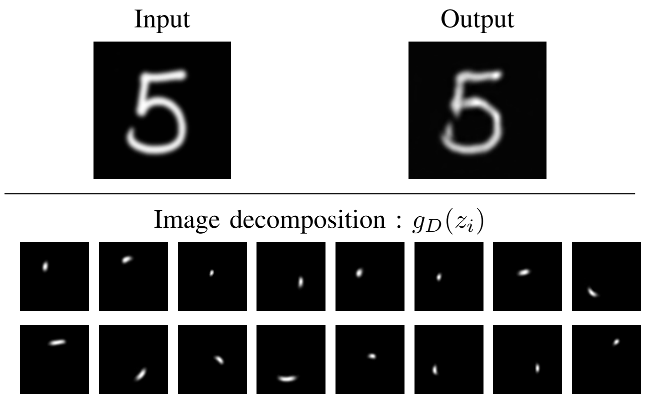

Fig. 2. **Decomposition of an mnist image.** The image of the number 5 has been broken down into several simple strokes, which are spatially localized.

are often non-convex functions that are generally NP-hard to calculate.

With a fixed autoencoder parametrized by a leaky ReLU DNN, complex data can only be represented on a bounded latent set in $\mathbb{R}^{kd_0}$. We see this by noticing that such an architecture can be seen as a piecewise affine functions. In particular, the width of unbounded affine regions necessarily grows when the norm of the code increases to infinity. Determining the bounds of the latent set is an open question in itself. In our examples we suppose that they can be well estimated by looking at the bounds of the latent codes of the training data $f_E(x_i)$.

Of course, the question of determining the size of the latent code is important in itself. This question is outside the scope of this paper, but ideas such as sparse regularization of the atomic latent code may be useful in estimating these parameters. Also, the properties of atomic autoencoders show that d_0 must be chosen small enough so that atoms cannot be "split" into smaller atoms.

We give practical implementation details for the experiments in the supplementary material. We draw attention to the fact that the resulting architecture is simple, with around 300,000 parameters (depending on the exact application), which is an extremely lightweight network.

III. Applications

We train an atomic autoencoder in two different examples and show that it yields low-dimensional representations with *atomic disantanglement*. To the best of our knowledge, no other existing generative DNN architecture yields such properties without additional constraints during learning.

A. Application to images consisting of parametric curves

We first consider images x consisting of a set of parametric curves: we suppose $x = \sum_{i=1}^{K} S(\theta_i)$, where $S(\theta_i)$ is a parametric curve, with parameters θ_i (e.g. a stroke parametrized by Bezier curves). We apply this modelling to the MNIST database, which is a good fit since each number in mnist is an amalgamation of small penstrokes, which are approximately

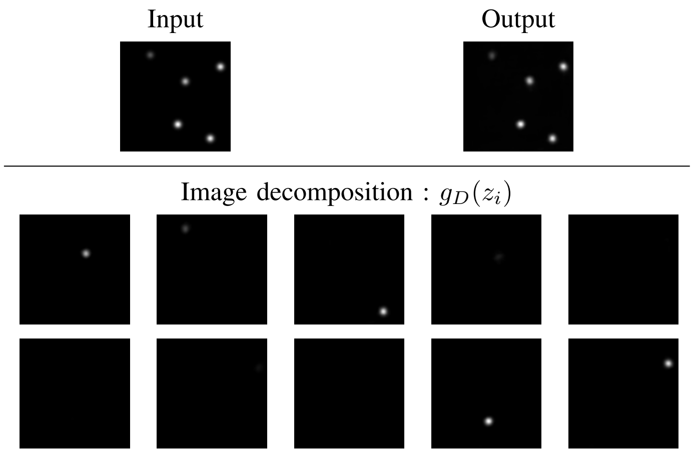

Fig. 3. **Decomposition of an image of off-the-grid spikes convolved with a Gaussian using 10 blocks of latent codes of dimension $d_0 = 3$.** Out atomic autoencoder separates spikes in each latent code block, even though our method is wholly unsupervised. This is due to its atomic structure.

simple splines convolved with the shape of the tip of the pen. We use a high-resolution version of MNIST [2] (500×500), which we downscale to 128×128. The final latent space size is 96 (block size $d_0 = 6$, and $k = 16$).

We verify that the autoencoder has indeed worked by showing the input and output in the supplementary material (with some distortion typical of raw autoencoder architectures). We show that the decomposition given by the model is meaningful in Figure 2. In this Figure, we show both the input and output of the network, and the individual decomposition images $g(z_i)$'s. We observe that the network successfully separates the different strokes. We also remark that the network does not mix up the spatial locations of the strokes: each one is connected and continuous. This is very satisfying behaviour, since at no point have we shown any such examples to the autoencoder: it learns these simple atoms on its own. In the supplementary material, we perform a linear interpolation in the latent space between two numbers. We verify that each point of the interpolation does not leave the space of images that are sum of parametric curves.

B. Application to off-the-grid sparse spike modelling

An important inverse problem with applications in microscopy, astronomy or acoustic signal processing is spike detection in images. This corresponds to the task of estimating positions and amplitudes of a series of *spikes* in an image convolved with a filter. We create a synthetic database where we allow for a maximum of 10 spikes, which are convolved with a Gaussian filter. The spikes have minimum separation larger than the standard deviation of the Gaussian filter. In this situation, we know that the size of latent blocks is $d_0 = 3$ (position and amplitude). We show in Figure 3 the application of the trained atomic autoencoder. Again, the network has learned, with no supervision, to separate the spikes in each image $g(z_i)$. Futhermore, in Figure 4, we see an example of navigation in the latent space. We have chosen a certain

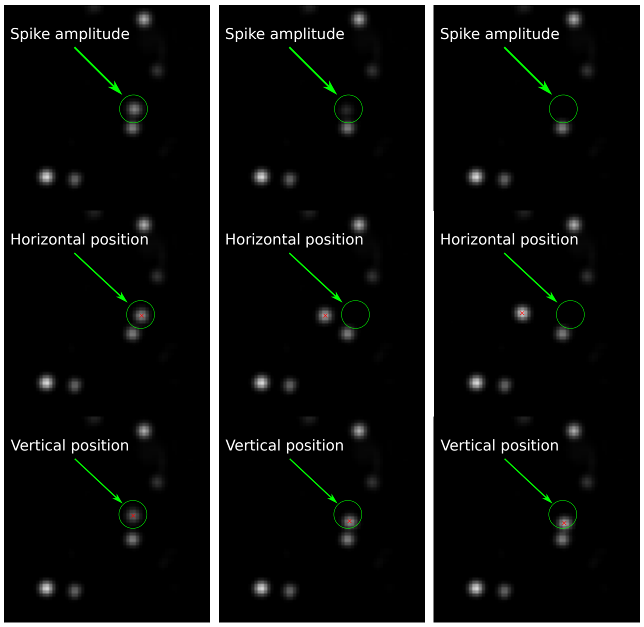

Fig. 4. **Navigation in the latent space for images of spikes.** Green: original spike. Left to right: linearly modification of a latent coordinate in a block, keeping the others constant. Top to bottom: modification of the three different coordinates of the latent block. The first coordinate changes the amplitude, while the second and third modify the position (red mark). No supervision in the training was involved here.

block i and then modified each latent coordinate of z_i. This modifies the behaviour of the third spike from the top of the image. The network codes the amplitude of the spike in the first coordinate, then two perpendicular motions in the next two. With a conventional architecture it could have mixed amplitude and positions. Finally, we observe that the motion is faster in one direction than the other; indeed there is nothing which imposes them to be the same.

Acknowledgements: A. Newson acknowledges the support of the French Agence Nationale de la Recherche (ANR) under reference ANR-21-CE23-0024 IDEGEN. Y. Traonmilin acknowledges the support of the French Agence Nationale de la Recherche (ANR) under reference ANR-20-CE40-0001 EFFIREG.

IV. Conclusion

We have introduced a simple architecture for the design of autoencoders induced by the theory of low-dimensional models. We gave elementary qualitative properties about the behaviour of this architecture. We show in two different applications how atomic autoencoders yields disentangled low-level interpretable parametetric representations.

Many questions arise from these works. Can this architecture achieve high level disentangled semantic representations in images? Could we further normalize latent blocks to ease navigation in the latent space? Many applications could benefit from this architecture and adapting it to their specificities is an open line of work. On the theoretical side, what exact behaviour can we expect from atomic autoencoders when they learn the data model up to a given distortion? In order to better understand the disentanglement properties of atomic autoencoders, can we further expand the notion of functions ψ that cannot be broken down into simpler functions?

References

[1] Ackley, D.H., Hinton, G.E., Sejnowski, T.J.: A learning algorithm for boltzmann machines. Cognitive Science **9**(1), 147–169 (1985)
[2] Beaulac, C., Rosenthal, J.S.: Introducing a new high-resolution handwritten digits data set with writer characteristics. SN Computer Science **4**(1), 1–12 (2023)
[3] Berner, J., Grohs, P., Voigtlaender, F.: Training relu networks to high uniform accuracy is intractable. arXiv preprint (2022)
[4] Bourlard, H., Kamp, Y.: Auto-association by multilayer perceptrons and singular value decomposition. Biological Cybernetics **59**(4), 291–294 (1988)
[5] Burgess, C.P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., Lerchner, A.: Understanding disentangling in β-VAE. arXiv:1804.03599 (2018)
[6] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: InfoGAN: interpretable representation learning by information maximizing generative adversarial nets. Advances in Neural Information Processing Systems pp. 2180–2188 (2016)
[7] Cheung, B., Livezey, J.A., Bansal, A.K., Olshausen, B.A.: Discovering Hidden Factors of Variation in Deep Networks. In: 3rd International Conference on Learning Representations, ICLR 2015, Workshop Track Proceedings (2015)
[8] Elman, J.L., Zipser, D.: Learning the hidden structure of speech. The Journal of the Acoustical Society of America **83**(4), 1615–1626 (1988)
[9] Foucart, S., Rauhut, H.: A mathematical introduction to compressive sensing. Springer (2013)
[10] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., Lerchner, A.: beta-vae: Learning basic visual concepts with a constrained variational framework (2016)
[11] Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural computation **3**(1), 79–87 (1991)
[12] Kim, H., Mnih, A.: Disentangling by Factorising. Proceedings of the 35th International Conference on Machine Learning pp. 2649–2658 (2018)
[13] Kumar, A., Sattigeri, P., Balakrishnan, A.: Variational Inference of Disentangled Latent Concepts from Unlabeled Observations (2018)
[14] Lample, G., Zeghidour, N., Usunier, N., Bordes, A., Denoyer, L., Ranzato, M.A.: Fader Networks:Manipulating Images by Sliding Attributes. Advances in Neural Information Processing Systems **30** (2017)
[15] Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature **401**(6755), 788–791 (1999)
[16] Lezama, J.: Overcoming the Disentanglement vs Reconstruction Tradeoff via Jacobian Supervision. 7th International Conference on Learning Representations (2019)
[17] Peng, P., Jalali, S., Yuan, X.: Solving inverse problems via autoencoders. IEEE Journal on Selected Areas in Information Theory **1**(1), 312–323 (2020)
[18] Tariyal, S., Majumdar, A., Singh, R., Vatsa, M.: Deep dictionary learning. IEEE Access **4**, 10096–10109 (2016)
[19] Tošić, I., Frossard, P.: Dictionary learning. IEEE Signal Processing Magazine **28**(2), 27–38 (2011)
[20] Wang, Y., Hua, Y., Candés, E., Pilanci, M.: Overparameterized relu neural networks learn the simplest models: Neural isometry and exact recovery (2022)