The Multi-Entity Variational Autoencoder

Charlie Nash[1,2]*, Ali Eslami[2], Chris Burgess[2], Irina Higgins[2],
Daniel Zoran[2], Theophane Weber[2], Peter Battaglia[2]
[1]Edinburgh University [2]DeepMind

Abstract

Representing the world as objects is core to human intelligence. It is the basis of people's sophisticated capacities for reasoning, imagining, planning, and learning. Artificial intelligence typically assumes human-defined representations of objects, and little work has explored how object-based representations can arise through unsupervised learning. Here we present an approach for learning probabilistic, object-based representations from data, called the "multi-entity variational autoencoder" (MVAE), whose prior and posterior distributions are defined over a *set* of random vectors. We demonstrate that the model can learn interpretable representations of visual scenes that disentangle objects and their properties.

1 Introduction

Human intelligence is object-oriented. Infants begin parsing the world into distinct objects within their first months of life [12], and our sophisticated capacities for reasoning, imagining, planning, and learning depend crucially on our representation of the world as dynamic objects and their relations. Though the human notion of an object is rich, and exists in an even richer continuum of non-solids, non-rigids, object parts, and multi-object configurations, here we use the term "object" simply as a discrete visual entity localized in space.

Many important domains of artificial intelligence use representations of objects that were chosen ahead of time by humans, based on subjective knowledge of what constitutes an object (e.g. patches in images that can be categorized, or geometric meshes for physical control). This core object knowledge was learned through evolution and experience, and is very useful. It allows can be shared across object instances, provides a means for some properties of the world to be highly dependent and others to be relatively independent, and allows objects to be composed to form abstractions and hierarchies whose wholes are greater than the sums of their parts. Given the importance of such representations, and the high cost of manually translating it from the engineer's mind into AI datasets and architectures, this work asks: How can an artificial system learn, without supervision, an object-based representation?

Our contributions are: (1) a probabilistic model that can learn object-based representations from data, (2) a visual attention mechanism for inferring a sparse set of objects from images.

2 Multi-entity VAE

The multi-entity VAE (MVAE) is a latent variable model of data $\mathbf{x}$ in which the latent space is factored into a set of N independent 'object' representations $\{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$. The MVAE defines a generative process in which each $\mathbf{z}_n$ $(n = 1, ..., N)$ is sampled independently from a prior distribution, $p(\mathbf{z})$, and data examples are sampled from a decoder distribution $p(\mathbf{x}|\mathbf{z})$.

*Work done during an internship at DeepMind

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

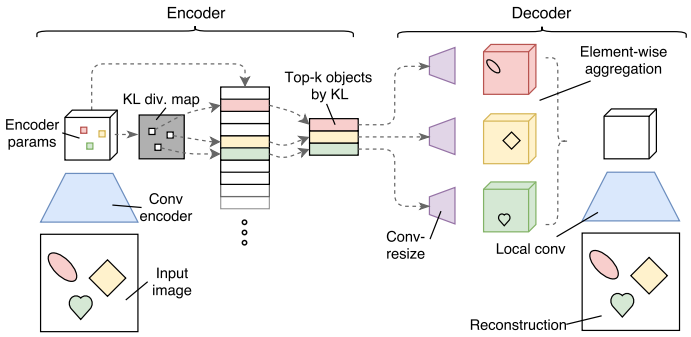


Figure 1: **Multi-entity VAE.** The encoder takes input images and produces a spatial map of posterior parameters. The KL-divergence of the posterior distribution against the prior is computed in each spatial location. The top-N posterior parameters by KL-divergence are selected from the spatial map, removing the spatial structure. These distributions are sampled, and the samples are passed independently through a shared convolution / upsampling network. The resulting object feature maps are aggregated using an element-wise operation, and a final convolutional network produces the output parameters.

In our visual experiments, the MVAE model assumes $p(\mathbf{z}_n)$ is a D-dimensional Gaussian with zero mean and unit variance. The conditional data distribution is implemented as a three-step deterministic decoding function, $\mathbf{f}$, which first maps each latent object representation to a processed object representation using a shared function, aggregates the processed object representations together, and deterministically transforms the result into the parameters of a Bernoulli distribution over pixel values. Crucially, $\mathbf{f}$ is permutation invariant with respect to the set of object representations. This encourages the model to learn object representations that are consistent and interchangeable.

Shared object processing. In the first stage of the decoder a shared function $\mathbf{g} : \mathbb{R}^D \to \mathbb{R}^K$ is applied independently to each latent object representation, resulting in a set of processed object descriptions $\mathbf{o}_n = \mathbf{g}(\mathbf{z}_n)$, $n = 1, \ldots, N$. These deterministic transformations of the prior latent variables are themselves random variables, which have dependencies induced by the prior latents. The K-dimensional object descriptions could be of any shape, but in this work we used 3D tensors as a structural bias towards representations of visual attributes. We implement $\mathbf{g}$ as a network that transforms each latent vector to a 3D tensor via reshaping, convolutions and upsampling.

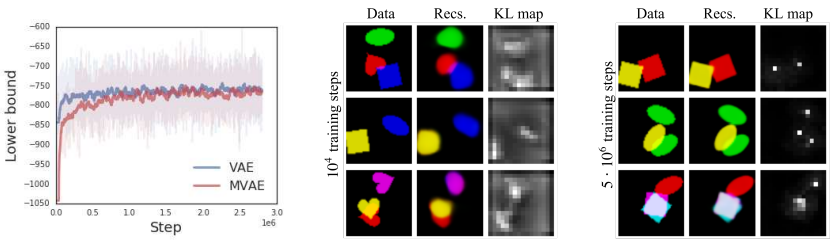
Aggregation. The processed object descriptions $\mathbf{o}_{1:N}$ are aggregated using a symmetric pooling function, to form $\mathbf{o}_{\text{pool}}$, a tensor with the same shape as each of $\mathbf{o}_{1:N}$. In our experiments we used element-wise sum or max as aggregation functions.

Rendering. After pooling, the resulting $\mathbf{o}_{\text{pool}}$ is mapped (i.e. rendered) to the element-wise parameters of the decoder distribution $\boldsymbol{\theta} = \mathbf{h}(\mathbf{o}_{\text{pool}})$. In our experiments $\mathbf{o}_{\text{pool}}$ is a 3D tensor, and $\mathbf{h}$ is a convolutional, upsampling network which outputs pixel-wise Bernoulli logits.

2.1 Maximal information attention

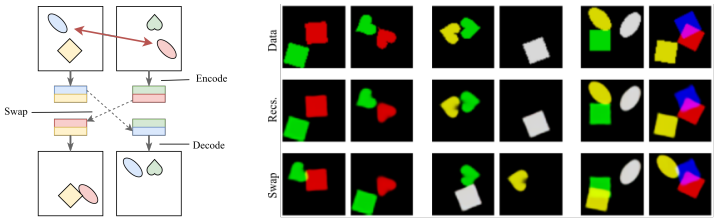
We employ amortized variational inference and learn a parameterized approximate posterior $q(\mathbf{z}_n|\mathbf{x})$ for each latent object representation. Unlike prior work [3, 4], we do not employ a learned attention mechanism in order to localise objects, but instead generate a large collection of candidate object inferences, from which N objects are selected. This inference method has the advantage that it circumvents the need for an explicitly learned attention mechanism, which may require a large number of recurrent passes over the image. This enables us to model scenes with large numbers of objects, something that was challenging in prior work.

Candidate generation. We generate candidate object inferences for visual scenes using a convolutional-network which maps input images to a grid of posterior parameters. Each spatial location in this output feature map is treated as an object, and we perform candidate sub-selection of this set as described in the next section. After sub-selection the spatial structure present in the convolutional grid is destroyed, so we tag each object with its relative spatial coordinates at an intermediate feature map in the convolutional network.



(a) (b)

Figure 2: **Training.** (a) Training curves with exponential moving averages for the MVAE and a standard convolution VAE. (b) Model reconstructions and KL-divergence spatial maps at early and late stages of training. The KL maps are diffuse early in training and become increasingly sharp during optimization.



(a) (b)

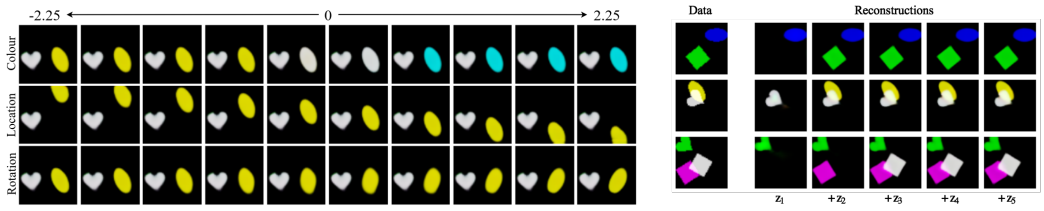
Figure 3: **Entity exchange.** (a) For a pair of input images, we encode to a set of latent objects before exchanging the representations of one objects in each pair. (b) Input pairs, model reconstructions, and reconstruction with exchanged entities. Here the objects in each input image with highest KL divergence are swapped.

Candidate sub-selection. Given a collection of candidate posteriors $\{q(\mathbf{z}_s|\mathbf{x})\}_s$ we compute the KL divergence $D^s_{\text{kl}} = D_{\text{kl}}[q(\mathbf{z}_s|\mathbf{x})|p(\mathbf{z})]$ for each candidate s. Approximate posteriors for the N latent objects are obtained by choosing the top-N objects by KL divergence. The intuition for this process is as follows: In order to reconstruct the input the network must encode lots of information in locations where objects exist like their shapes, colours, etc., whereas much less information is needed to encode background information; simply that there is no object present there. As such the 'object' and 'non-object' locations will have high and low KL-divergence respectively, and by choosing the top locations by KL-divergence we encode information only in the most informative regions of the image. We call this process maximal-information attention, and note that it can be used for any data modality where a superset of candidate inferences can be generated.

The candidate generation and sub-selection results in approximate posteriors for the N object representations, which we then sample from and pass to the decoder as in a standard VAE.

3 Related work

Our MVAE builds on previous neural probabilistic generative models, especially variational autoencoders (VAEs) [6, 10]. The DC-IGN [7], beta-VAE [5], and InfoGAN [1] are extensions and alternatives that promote learning latent codes whose individual features are "disentangled" [13, 2], i.e. correlated exclusively with underlying axes in the data-generating process. Other work has developed attention mechanisms for sampling subsets of visual scenes [9, 4, 14], which promotes learned representations that are spatially localized. Recurrent neural networks have been used in combination with spatial attention to allow for unsupervised learning of object locations and counts [3]. And several recent approaches allow object-like representations to be learned in supervised settings for visual question-answering and physical prediction [11, 15].



(a) (b)

Figure 4: (a)**Within-object latent traversals.** Decoded scenes for traversals from -2.25 to 2.25 for a selection of latent dimensions for a single object. The latent variables associated with different objects are held fixed during the process. (b) **Decoding objects sequentially.** (left) Data examples. (right) Reconstructions of the data where one latent object is added at each step. Here the latent objects $\mathbf{z}_1, \ldots, \mathbf{z}_5$ are ordered by the KL-divergence of the encoding distributions.

4 Experiments

We evaluate our model on a multiple object variant of the dSprites dataset [8]. This dataset consists of 64×64 images of sprite shapes with random colours, orientations and locations. Figure 2 shows training curves, model reconstructions and the development of KL-divergence attention maps over the course of training. For more experimental detail including model architectures see appendix A.

4.1 Between-object disentangling

In order to check whether the MVAE has learned a factored representation of objects we use the following qualitative tasks:

Entity exchange. One of the main motivations for learning disentangled representations of objects in a scene is that it facilitates compositional reasoning. We should be able to imagine an object in different contexts, independent of the original context in which we observed it. We examine our model's capacity to transfer objects into new contexts by encoding a pair of scenes, swapping the representations for one one object in each of the scenes, and then decoding both of the altered scenes. Figure 3 shows some example results for the MVAE. We note that entire objects are cleanly swapped between scenes, even in the presence of overlap or clutter, indicating that objects in the input image are cleanly partitioned into separate object representations.

Sequential decoding. Another qualitative indicator of the representations learned by the MVAE is to encode an input scene, then decode one object a time. As the MVAE's decoder performs object-wise aggregation, we can decode variable numbers of latent object representations. Figure 4a shows example decoded scenes in which we sequentially add one latent object representation at a time to the decoder. At each step a single object is introduced to the scene until all the objects present in the input scene have been decoded, and beyond this point the reconstructions are unaltered by the additional latent object representations. This indicates that the surplus latent object slots encode a 'Null' object, which decodes to nothing.

4.2 Within-object disentangling

To investigate the extent to which MVAE learns a representation that is disentangled within particular objects, we perform latent traversals for one object at a time, while keeping the other latent variables fixed. If the model has been successful we should expect to see that the underlying generative factors of the data are captured by single latent variables. Figure 4b shows a number of example latent traversals. The figure shows that the MVAE achieves a good degree of disentangling, e.g. with location factored from colour. This contrasts with the representations learned by a standard VAE (Appendix B), which are entangled across objects, with latent traversals causing multiple objects to deform and change colour simultaneously.

4.3 Unsupervised object counting

Here we demonstrate that the MVAEs spatial KL-divergence maps are a good proxy for object counting. We searched over KL-divergence thresholds on a training set of size 6400, and using the best training threshold tested on a 12800 newly sampled data examples. The chosen threshold gets 74.4% and 72.6% object count accuracy on the training and test sets respectively.

5 Discussion

Here we introduced a probabilistic model for learning object-based representations of visual scenes, which performs efficient inference using a novel one-shot informational attention mechanism that scales to large numbers of objects. Our results showed that our MVAE model can learn discrete, self-contained, interchangeable representations of multiple objects in a scene. We also found that the learned latent features of each object representation formed a disentangled code that separated the underlying factors of variation.

One limitation of this work is that the latent objects are assumed to be independent, which is obviously inconsistent with the tremendous statistical structure among objects in real scenes. Future work will tackle the task of learning not just about objects, but about their relations, interactions, and hierarchies in a unsupervised setting.

This work opens new directions for learning efficient and useful representations of complex scenes that may benefit question-answering and image captioning systems, as well as reinforcement learning agents. Moreover, this work may also help cognitive science explain why object-based representations are central to biological intelligence, as well as their evolutionary and experiential origins.

References

[1] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*, pages 2172–80, 2016.

[2] J. J. DiCarlo and D. D. Cox. Untangling invariant object recognition. *Trends in cognitive sciences*, 11(8):333–341, 2007.

[3] S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, K. Kavukcuoglu, and G. E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *NIPS*, pages 3225–3233, 2016.

[4] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra. DRAW: A recurrent neural network for image generation. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1462–1471. JMLR.org, 2015.

[5] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2016.

[6] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[7] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum. Deep convolutional inverse graphics network. In *NIPS*, pages 2539–2547, 2015.

[8] L. Matthey, I. Higgins, D. Hassabis, and A. Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

[9] V. Mnih, N. Heess, A. Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014.

[10] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[11] A. Santoro, D. Raposo, D. G. T. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. P. Lillicrap. A simple neural network module for relational reasoning. *NIPS*, abs/1706.01427, 2017.

[12] E. S. Spelke and K. D. Kinzler. Core knowledge. *Developmental science*, 10(1):89–96, 2007.

[13] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[15] N. Watters, A. Tacchetti, T. Weber, R. Pascanu, P. Battaglia, and D. Zoran. Visual interaction networks. *arXiv preprint arXiv:1706.01433*, 2017.

Appendix

A Experimental details

Data generation: Sprites are sampled independently, including whether or not they exist, up to some fixed upper bound on the number of sprites. We blend the colours of overlapping sprites by summing their RGB values and clipping at one. During training we generate these images dynamically, such that the MVAE always sees newly sampled data.

Encoder: The MVAE encoder is a convolutional network that starts with four convolutional layers with max-pooling. The resulting feature maps are concatenated with two channels of relative x,y co-ordinates followed by two further convolutional layers with a final channel dimensionality of 24. The output channels are split to form the means and log-variances of 12-dimensional diagonal Gaussian posteriors.

Decoder: Latent object representations are processed with a shared network $\mathbf{g}$. This network has two fully connected layers, the outputs of which are reshaped into a feature maps of shape $8 \times 8 \times 16$, followed by two convolutional layers with bilinear up-sampling. We aggregate across object representations using max-pooling, and then process the resulting feature map with a convolutional network $\mathbf{h}$. This network consists of three convolutional layers and one bilinear up-sampling layer. The network outputs a $64 \times 64 \times 3$ image of Bernoulli logits.

Standard VAE: We trained a baseline VAE with a comparable architecture to the MVAE. We match the latent dimensionality, using 60 latent dimensions. The encoder is identitical to the MVAE encoder, but the output feature maps are projected to the parameters of a 60-dimensional Gaussian using a linear layer. The decoder is equivalent to applying the shared object network $\mathbf{g}$ from the MVAE to the latent variables, and then passing the outputs into the rendering network $\mathbf{h}$.

Training: We train the MVAE using first-order gradient methods to maximize the variational lower bound on the log-probability of the data as in a standard VAE. All Models used elu non-linearities and were trained with Adam optimizer with scheduled learning rate annealing from 10^{-3} to 10^{-5} over the course of $3 * 10^6$ training steps.

B VAE latent traversals

As a comparison to the MVAE latent traversals in figure 4, we show traversals of latent dimensions for a trained VAE baseline in figure 5. Latent dimensions were hand-chosen as examples of variables that have a significant impact on the decoded results.

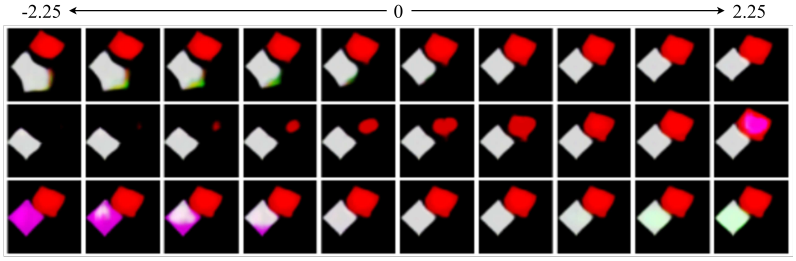


Figure 5: **VAE latent traversals.** Decoded scenes for traversals from -2.25 to 2.25 for a selection of latent dimensions for a standard VAE. All other latent variables are held fixed during the process.

C Reconstructions and samples

Figure 6 shows reconstructions for 3-sprite and 8-sprite datasets. We note that in the 3-sprite data that both the VAE and MVAE achieve good reconstructions, however the MVAE samples are of a higher quality, with more distinctly defined and coherent objects. The MVAE achieves good reconstructions and samples for the highly cluttered scenes in the 8-sprite data.

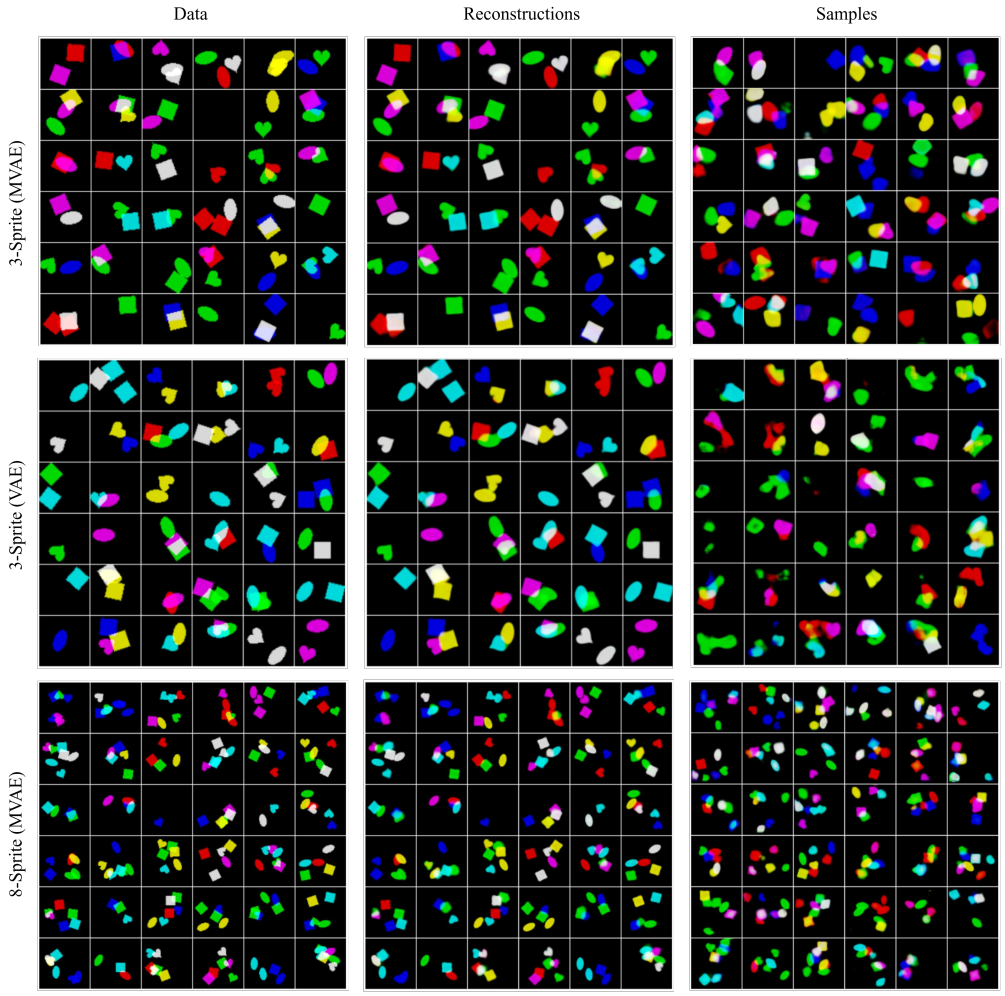


Figure 6: **Reconstructions and samples.** Data images, model reconstructions and model samples on the 3-sprite and 8-sprite datasets.

Adversarially Regularized Autoencoders for Unaligned Text Style-Transfer

Junbo (Jake) Zhao[1], Yoon Kim[2], Kelly Zhang[1], Alexander M. Rush[2], Yann LeCun[1,3]
[1] Department of Computer Science, New York University
[2] School of Engineering and Applied Sciences, Harvard University
[3] Facebook AI Research
{jakezhao,kz918,yann}@cs.nyu.edu, {yoonkim,srush}@seas.harvard.edu

Abstract

One general framework for learning disentangled representations is using a reconstruction loss to ensure a consistent mapping between the learned representations and the input, with an auxiliary loss that encourages disentanglement. While autoencoders for continuous structures, such as images or wave forms, have been quite successful, developing general-purpose autoencoders for discrete structures, such as text sequences, has proven to be more challenging. In particular, discrete inputs make it more difficult to learn a smooth encoder that preserves the complex local relationships in the input space. In this work, we propose an adversarially regularized autoencoder (ARAE) with the goal of learning more robust discrete-space representations. ARAE jointly trains both a rich discrete-space encoder and a simpler continuous space generator function, while using generative adversarial network (GAN) training to constrain the distributions to be similar. This method yields a smoother contracted code space that maps similar inputs to nearby codes, and also an implicit latent variable GAN model for generation. Experiments on text demonstrate that the autoencoder produces state-of-the-art results in unaligned text style transfer task using only a shared continuous-space representation.

1 Introduction

Recent work on regularized autoencoders, such as variational [6, 9] and denoising [11] variants, has shown significant progress in learning smooth representations of complex, high-dimensional continuous data such as images. Unfortunately, learning similar latent representations of discrete structures remains a challenging problem. Initial work on VAEs for text has shown that optimization is difficult, as the decoder can easily degenerate into a unconditional language model [2]. Recent work on generative adversarial networks (GANs) for text has mostly focused on getting around the use of discrete structures either through policy gradient methods [12] or with the Gumbel-Softmax distribution [7]. However, neither approach can yet produce robust representations directly.

A major difficulty of discrete autoencoders is mapping a discrete structure to a continuous code vector while also smoothly capturing the complex local relationships of the input space. Inspired by recent work combining pretrained autoencoders with deep latent variable models, we propose to target this issue with an adversarially regularized autoencoder (ARAE). Specifically we jointly train a discrete structure encoder and continuous space generator, while constraining the two models with a discriminator to agree in distribution. This approach allows us to utilize a complex encoder model and still constrain it with a very flexible, but more limited generator distribution. The full model can be then used as a smoother discrete structure autoencoder or as a latent variable GAN model where a sample can be decoded, with the same decoder, to a discrete output. Since the system produces a single continuous coded representation—in contrast to methods that act on each RNN state—it can easily be further regularized with problem-specific invariants, for instance to learn to ignore style, sentiment, or other attributes for transfer tasks.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.
∗ A full version of this paper is available at https://arxiv.org/pdf/1706.04223.pdf.

2 Background

Discrete Structure Autoencoders Define $\mathcal{X} = \mathcal{V}^n$ to be a set of discrete structures where $\mathcal{V}$ is the vocabulary, n is the sentence length, and $\mathbb{P}_x$ to be a distribution over this space. A discrete autoencoder consists of two parameterized functions: a deterministic encoder function $\text{enc}_\phi : \mathcal{X} \mapsto \mathcal{C}$ with parameters ϕ that maps from input to code space and a conditional decoder distribution $p_\psi(\mathbf{x} \mid \mathbf{c})$ over structures $\mathcal{X}$ with parameters ψ. The parameters are trained on a cross-entropy reconstruction loss:

$$\mathcal{L}_{\text{rec}}(\phi, \psi) = -\log p_\psi(\mathbf{x} \mid \text{enc}_\phi(\mathbf{x}))$$

The choice of the encoder and decoder parameterization is specific to the structure of interest, for example we use RNNs for sequences. We use the notation, $\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p_\psi(\mathbf{x} \mid \text{enc}_\phi(\mathbf{x}))$ for the (approximate) decoder mode. When $\mathbf{x} = \hat{\mathbf{x}}$ the autoencoder is said to perfectly reconstruct $\mathbf{x}$.

Generative Adversarial Networks GANs are a class of parameterized implicit generative models [4]. The method approximates drawing samples from a true distribution $\mathbf{c} \sim \mathbb{P}_r$ by instead employing a latent variable $\mathbf{z}$ and a parameterized deterministic generator function $\tilde{\mathbf{c}} = g_\theta(\mathbf{z})$ to produce samples $\tilde{\mathbf{c}} \sim \mathbb{P}_g$. Initial work on GANs minimizes the Jensen-Shannon divergence between the distributions. Recent work on Wasserstein GAN (WGAN) [1], replaces this with the *Earth-Mover* (Wasserstein-1) distance.

GAN training utilizes two separate models: a *generator* $g_\theta(\mathbf{z})$ that maps a $\mathbf{z}$ vector sampled from a distribution to a code-space vector and a critic/discriminator $f_w(\mathbf{c})$ that aims to distinguish *real* data and *generated* samples from g_θ. Informally, the generator is trained to fool the critic, and the critic is trained to tell real from generated codes. WGAN training uses the following min-max optimization over generator parameters θ and critic parameters w,

$$\min_\theta \max_{w \in \mathcal{W}} \mathbb{E}_{\mathbf{c} \sim \mathbb{P}_r}[f_w(\mathbf{c})] - \mathbb{E}_{\tilde{\mathbf{c}} \sim \mathbb{P}_g}[f_w(\tilde{\mathbf{c}})], \tag{1}$$

where $f_w : \mathcal{C} \mapsto \mathbb{R}$ denotes the critic function, $\tilde{\mathbf{c}}$ is obtained from the generator, $\tilde{\mathbf{c}} = g_\theta(\mathbf{z})$, and $\mathbb{P}_r$ and $\mathbb{P}_g$ are real and generated distributions. If the critic parameters w are restricted to an 1-Lipschitz function set $\mathcal{W}$, this term correspond to minimizing Wasserstein-1 distance $W(\mathbb{P}_r, \mathbb{P}_g)$. We use a naive approximation to enforce this property by weight-clipping, i.e. $w = [-\epsilon, \epsilon]^d$ [1].

3 Model: Adversarially Regularized Autoencoder

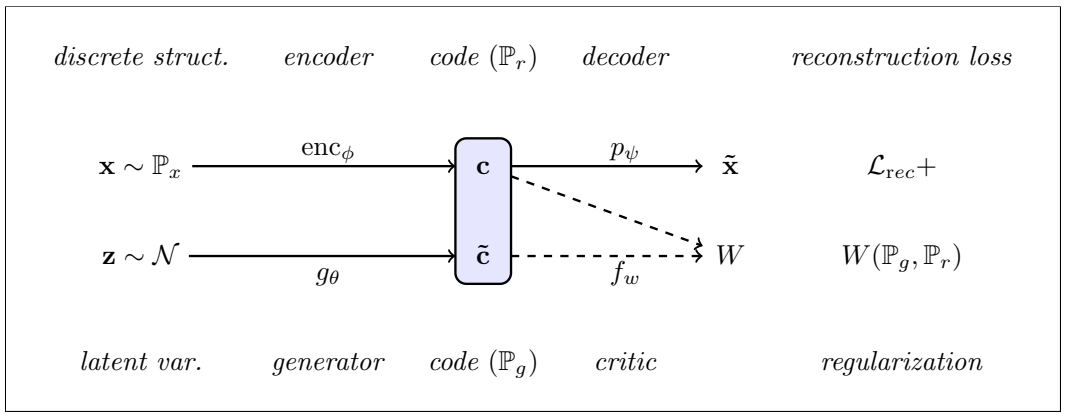


Figure 1: ARAE architecture. The model can be used as an autoencoder, where a structure $\mathbf{x}$ is encoded and decoded to produce $\hat{\mathbf{x}}$, and as a GAN (ARAE-GAN), where a sample $\mathbf{z}$ is passed though a generator g_θ to produce a code vector, which is similarly decoded to $\tilde{\mathbf{x}}$. The critic function f_w is only used at training to help approximate W.

Ideally, a discrete autoencoder should be able to *reconstruct* x from c, but also *smoothly* assign similar codes c and c' to similar x and x'. For continuous autoencoders, this property can be enforced directly through explicit regularization. How can we enforce that similar discrete structures map to nearby codes?

Adversarially regularized autoencoders target this issue by learning a parallel continuous-space generator with a restricted functional form to act as a smoother reference encoding. The joint objective regularizes the autoencoder to constrain the discrete encoder to agree in distribution with its continuous counterpart:

$$\min_{\phi,\psi,\theta} \quad \mathcal{L}_{\text{rec}}(\phi,\psi) + \lambda^{(1)} W(\mathbb{P}_r, \mathbb{P}_g)$$

Above W is the Wasserstein-1 distance between $\mathbb{P}_r$ the distribution of codes from the discrete encoder model ($\text{enc}_\phi(x)$ where $x \sim \mathbb{P}(x)$) and $\mathbb{P}_g$ is the distribution of codes from the continuous generator model ($g_\theta(z)$ for some z, e.g. $z \sim \mathcal{N}(0, I)$). To approximate Wasserstein-1 term, the W function includes an embedded critic function which is optimized adversarially to the encoder and generator as described in the background. The full model is shown in Figure 1.

To train the model, we use a block coordinate descent to alternate between optimizing different parts of the model: (1) the encoder and decoder to minimize reconstruction loss, (2) the WGAN critic function to approximate the W term, (3) the encoder and generator to adversarially fool the critic to minimize W:

$$\begin{aligned}
&1)\ \min_{\phi,\psi} && \mathcal{L}_{\text{rec}}(\phi,\psi) && \\
&2)\ \min_{w\in\mathcal{W}} && \mathcal{L}_{\text{cri}}(w) = && \max_{w\in\mathcal{W}} \quad \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_x}\left[f_w(\text{enc}_\phi(\mathbf{x}))\right] - \mathbb{E}_{\tilde{\mathbf{c}}\sim\mathbb{P}_g}\left[f_w(\tilde{\mathbf{c}})\right] \\
&3)\ \min_{\phi,\theta} && \mathcal{L}_{\text{encs}}(\phi,\theta) = && \min_{\phi,\theta} \quad \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_x}\left[f_w(\text{enc}_\phi(\mathbf{x}))\right] - \mathbb{E}_{\tilde{\mathbf{c}}\sim\mathbb{P}_g}\left[f_w(\tilde{\mathbf{c}})\right]
\end{aligned}$$

Algorithm 1 ARAE Training

for number of training iterations **do**

(1) Train the autoencoder for reconstruction $[\mathcal{L}_{\text{rec}}(\phi,\psi)]$.

Sample $\{\mathbf{x}^{(i)}\}_{i=1}^m \sim \mathbb{P}_x$ and compute code-vectors $\mathbf{c}^{(i)} = \text{enc}_\phi(\mathbf{x}^{(i)})$.

Backpropagate reconstruction loss, $\mathcal{L}_{\text{rec}} = -\frac{1}{m}\sum_{i=1}^m \log p_\psi(\mathbf{x}^{(i)} | \mathbf{c}^{(i)}, [\mathbf{y}^{(i)}])$, and update.

(2) Train the critic $[\mathcal{L}_{\text{cri}}(w)]$ (Repeat k times)

Sample $\{\mathbf{x}^{(i)}\}_{i=1}^m \sim \mathbb{P}_x$ and $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim \mathcal{N}(0, \mathbf{I})$.

Compute code-vectors $\mathbf{c}^{(i)} = \text{enc}_\phi(\mathbf{x}^{(i)})$ and $\tilde{\mathbf{c}}^{(i)} = g_\theta(\mathbf{z}^{(i)})$.

Backpropagate loss $-\frac{1}{m}\sum_{i=1}^m f_w(\mathbf{c}^{(i)}) + \frac{1}{m}\sum_{i=1}^m f_w(\tilde{\mathbf{c}}^{(i)})$, update, clip the critic w to $[-\epsilon, \epsilon]^d$.

(2b) Train the code classifier $[\min_u \mathcal{L}_{\text{class}}(\phi, u)]$

Sample $\{\mathbf{x}^{(i)}\}_{i=1}^m \sim \mathbb{P}_x$, lookup $y^{(i)}$, and compute code-vectors $\mathbf{c}^{(i)} = \text{enc}_\phi(\mathbf{x}^{(i)})$.

Backpropagate loss $-\frac{1}{m}\sum_{i=1}^m \log p_u(y^{(i)} | \mathbf{c}^{(i)})$, update.

(3) Train the generator and encoder adversarially to critic $[\mathcal{L}_{\text{encs}}(\phi,\theta)]$

Sample $\{\mathbf{x}^{(i)}\}_{i=1}^m \sim \mathbb{P}_x$ and $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim \mathcal{N}(0, \mathbf{I})$

Compute code-vectors $\mathbf{c}^{(i)} = \text{enc}_\phi(\mathbf{x}^{(i)})$ and $\tilde{\mathbf{c}}^{(i)} = g_\theta(\mathbf{z}^{(i)})$.

Backpropagate adversarial loss $\frac{1}{m}\sum_{i=1}^m f_w(\mathbf{c}^{(i)}) - \frac{1}{m}\sum_{i=1}^m f_w(\tilde{\mathbf{c}}^{(i)})$ and update.

(3b) Train the encoder adversarially to code classifier $[\max_\phi \mathcal{L}_{\text{class}}(\phi, u)]$

Sample $\{\mathbf{x}^{(i)}\}_{i=1}^m \sim \mathbb{P}_x$, lookup $y^{(i)}$, and compute code-vectors $\mathbf{c}^{(i)} = \text{enc}_\phi(\mathbf{x}^{(i)})$.

Backpropagate adversarial classifier loss $-\frac{1}{m}\sum_{i=1}^m \log p_u(1 - y^{(i)} \mid \mathbf{c}^{(i)})$, update.

Code Space Transfer One benefit of the ARAE framework is that it compresses the input to a single code vector. This framework makes it ideal for manipulating discrete objects while in continuous code space. For example, consider the problem of unaligned transfer, where we want to change an attribute of a discrete input without supervised examples, e.g. to change the topic or sentiment of a sentence. First, we extend the decoder to condition on a transfer variable denoting this

	Automatic Evaluation				Human Evaluation		
Model	Transfer	BLEU	PPL	Reverse PPL	Transfer	Similarity	Naturalness
Cross-Aligned AE	77.1%	17.75	65.9	124.2	57%	3.8	2.7
AE	59.3%	37.28	31.9	68.9	-	-	-
ARAE, $\lambda_a^{(1)}$	73.4%	31.15	29.7	70.1	-	-	-
ARAE, $\lambda_b^{(1)}$	81.8%	20.18	27.7	77.0	74%	3.7	3.8

Table 1: Experiments on sentiment transfer. Left shows the automatic metrics (Transfer/BLEU/PPL/Reverse PPL); right shows human evaluation metrics (Transfer/Similarity/Naturalness). Cross-Aligned AE is from [10].

	Positive ⇒ Negative		Negative ⇒ Positive
	great indoor mall .		hell no !
ARAE	no smoking mall .	ARAE	hell great !
Cross-AE	terrible outdoor urine .	Cross-AE	incredible pork !
	it has a great atmosphere , with wonderful service .		small , smokey , dark and rude management .
ARAE	it has no taste , with a complete jerk .	ARAE	small , intimate , and cozy friendly staff .
Cross-AE	it has a great horrible food and run out service .	Cross-AE	great , , , chips and wine .
	we came on the recommendation of a bell boy and the food was amazing .		the people who ordered off the menu did n't seem to do much better .
ARAE	we came on the recommendation and the food was a joke .	ARAE	the people who work there are super friendly and the menu is good .
Cross-AE	we went on the car of the time and the chicken was awful .	Cross-AE	the place , one of the office is always worth you do a business .

Table 2: Sentiment transfer results. Original sentence and transferred output (from ARAE and the Cross-Aligned AE) of 6 randomly-drawn examples.

	Original Science		Original Music		Original Politics
	what is an event horizon with regards to black holes ?		do you know a website that you can find people who want to join bands ?		republicans : would you vote for a cheney / satan ticket in 2008 ?
Music	what is your favorite sitcom with adam sandler ?	Science	do you know a website that can help me with science ?	Science	guys : how would you solve this question ?
Politics	what is an event with black people ?	Politics	do you think that you can find a person who is in prison ?	Music	guys : would you rather be a good movie ?
	take 1ml of hcl (concentrated) and dilute it to 50ml .		all three are fabulous artists , with just incredible talent ! !		4 years of an idiot in office + electing the idiot again = ?
Music	take em to you and shout it to me	Science	all three are genetically bonded with water , but just as many substances , are capable of producing a special case .	Science	4 years of an idiot in the office of science ?
Politics	take bribes to islam and it will be punished .	Politics	all three are competing with the government , just as far as i can .	Music	4) <unk> in an idiot , the idiot is the best of the two points ever !
	just multiply the numerator of one fraction by that of the other .		but there are so many more i can 't think of !		anyone who doesnt have a billion dollars for all the publicity cant win .
Music	just multiply the fraction of the other one that 's just like it .	Science	but there are so many more of the number of questions .	Science	anyone who doesnt have a decent chance is the same for all the other .
Politics	just multiply the same fraction of other countries .	Politics	but there are so many more of the can i think of today .	Music	anyone who doesnt have a lot of the show for the publicity .

Table 3: Random samples from Yahoo topic transfer. Note the first row is from ARAE trained on titles while the following ones are from replies.

attribute $\mathbf{y}$ which is known during training, to learn $p_\psi(\mathbf{x} \mid \mathbf{c}, y)$. Next, we train the code space to be invariant to this attribute, to force it to be learned fully by the decoder. Specifically, we further regularize the code space to map similar x with different attribute labels y near enough to fool a code space attribute classifier, i.e.:

$$\min_{\phi,\psi,\theta} \quad \mathcal{L}_{\text{rec}}(\phi,\psi) + \lambda^{(1)} W(\mathbb{P}_r, \mathbb{P}_g) - \lambda^{(2)} \mathcal{L}_{\text{class}}(\phi, u)$$

where $\mathcal{L}_{\text{class}}(\phi, u)$ is the loss of a classifier $p_u(y \mid \mathbf{c})$ from code space to labels (in our experiments we always set $\lambda^{(2)} = 1$). To incorporate this additional regularization, we simply add two more gradient update steps: (2b) training a classifier to discriminate codes, and (3b) adversarially training the encoder to fool this classifier. Note that similar techniques have been introduced in other domains, notably in images [8] and video modeling [3].

4 Experiments: Unaligned Text Transfer

For sentiment we follow the same setup as [10] and split the Yelp corpus into two sets of unaligned positive and negative reviews. We train an ARAE as an autoencoder with two separate decoders, one for positive and one for negative sentiment, and incorporate adversarial training of the encoder to remove sentiment information from the code space. We test by encoding in sentences of one class and decoding, greedily, with the opposite decoder.

Our evaluation is based on four automatic metrics, shown in Table 1: (i) Transfer: measuring how successful the model is at transferring sentiment based on an automatic classifier (we use the `fastText` library [5]). (ii) BLEU: measuring the consistency between the transferred text and the original. We expect the model to maintain as much information as possible and transfer only the style; (iii) Perplexity: measuring the fluency of the generated text; (iv) Reverse Perplexity: measuring the extent to which the generations are representative of the underlying data distribution.[1] Both perplexity numbers are obtained by training an RNN language model.

[1]This *reverse perplexity* is calculated by training a language model on the generated data and measuring perplexity on held-out, real data (i.e. reverse of regular perplexity). We also found this metric to be helpful for early-stopping based on validation data.

We additionally perform human evaluations on the cross-aligned AE and our best ARAE model. We randomly select 1000 sentences (500/500 positive/negative), obtain the corresponding transfers from both models, and ask Amazon Mechanical Turkers to evaluate the sentiment (Positive/Neutral/Negative) and naturalness (1-5, 5 being most natural) of the transferred sentences. We create a separate task in which we show the Turkers the original and the transferred sentences, and ask them to evaluate the similarity based on sentence structure (1-5, 5 being most similar). We explicitly ask the Turkers to disregard sentiment in their similarity assessment.

The same method can be applied to other style transfer tasks, for instance the more challenging Yahoo QA data [13]. For Yahoo we chose 3 relatively distinct topic classes for transfer: Science & Math, Entertainment & Music, and Politics & Government. As the dataset contains both questions and answers, we separated our experiments into titles (questions) and replies (answers). The qualitative results are showed in table 3. See Appendix 5 for additional generation examples.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv:1701.07875*, 2017.

[2] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

[3] Emily Denton and Vighnesh Birodkar. Unsupervised learning of disentangled representations from video. *arXiv preprint arXiv:1705.10915*, 2017.

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of NIPS*, 2014.

[5] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[6] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *Proceedings of ICLR*, 2014.

[7] Matt Kusner and Jose Miguel Hernandez-Lobato. GANs for Sequences of Discrete Elements with the Gumbel-Softmax Distribution. *arXiv:1611.04051*, 2016.

[8] Guillaume Lample, Neil Zeghidour, Nicolas Usuniera, Antoine Bordes, Ludovic Denoyer, and Marc'Aurelio Ranzato. Fader networks: Manipulating images by sliding attributes. In *Proceedings of NIPS*, 2017.

[9] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of ICML*, 2014.

[10] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style Transfer from Non-Parallel Text by Cross-Alignment. *arXiv:1705.09655*, 2017.

[11] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of ICML*, 2008.

[12] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *Proceedings of AAAI*, 2017.

[13] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

5 Appendix: Sheet of style-transfer samples

Yelp Transfer

	Positive to Negative		Negative to Positive
Original	great indoor mall .	Original	hell no !
ARAE	no smoking mall .	ARAE	hell great !
Cross-AE	terrible outdoor urine .	Cross-AE	incredible pork !
Original	great blooming onion .	Original	highly disappointed !
ARAE	no receipt onion .	ARAE	highly recommended !
Cross-AE	terrible of pie .	Cross-AE	highly clean !
Original	i really enjoyed getting my nails done by peter .	Original	bad products .
ARAE	i really needed getting my nails done by now .	ARAE	good products .
Cross-AE	i really really told my nails done with these things .	Cross-AE	good prices .
Original	definitely a great choice for sushi in las vegas !	Original	i was so very disappointed today at lunch .
ARAE	definitely a _num_ star rating for _num_ sushi in las vegas .	ARAE	i highly recommend this place today .
Cross-AE	not a great choice for breakfast in las vegas vegas !	Cross-AE	i was so very pleased to this .
Original	the best piece of meat i have ever had !	Original	i have n't received any response to anything .
ARAE	the worst piece of meat i have ever been to !	ARAE	i have n't received any problems to please .
Cross-AE	the worst part of that i have ever had had !	Cross-AE	i have always the desert vet .
Original	really good food , super casual and really friendly .	Original	all the fixes were minor and the bill ?
ARAE	really bad food , really generally really low and decent food .	ARAE	all the barbers were entertaining and the bill did n't disappoint .
Cross-AE	really good food , super horrible and not the price .	Cross-AE	all the flavors were especially and one !
Original	it has a great atmosphere , with wonderful service .	Original	small , smokey , dark and rude management .
ARAE	it has no taste , with a complete jerk .	ARAE	small , intimate , and cozy friendly staff .
Cross-AE	it has a great horrible food and run out service .	Cross-AE	great , , , chips and wine .
Original	their menu is extensive , even have italian food .	Original	the restaurant did n't meet our standard though .
ARAE	their menu is limited , even if i have an option .	ARAE	the restaurant did n't disappoint our expectations though .
Cross-AE	their menu is decent , i have gotten italian food .	Cross-AE	the restaurant is always happy and knowledge .
Original	everyone who works there is incredibly friendly as well .	Original	you could not see the stage at all !
ARAE	everyone who works there is incredibly rude as well .	ARAE	you could see the difference at the counter !
Cross-AE	everyone who works there is extremely clean and as well .	Cross-AE	you could definitely get the fuss !
Original	there are a couple decent places to drink and eat in here as well .	Original	room is void of all personality , no pictures or any sort of decorations .
ARAE	there are a couple slices of options and _num_ wings in the place .	ARAE	room is eclectic , lots of flavor and all of the best .
Cross-AE	there are a few night places to eat the car here are a crowd .	Cross-AE	it 's a nice that amazing , that one 's some of flavor .
Original	if you 're in the mood to be adventurous , this is your place !	Original	waited in line to see how long a wait would be for three people .
ARAE	if you 're in the mood to be disappointed , this is not the place .	ARAE	waited in line for a long wait and totally worth it .
Cross-AE	if you 're in the drive to the work , this is my place !	Cross-AE	another great job to see and a lot going to be from dinner .
Original	we came on the recommendation of a bell boy and the food was amazing .	Original	the people who ordered off the menu did n't seem to do much better .
Cross-AE	we came on the recommendation and the food was a joke .	ARAE	the people who work there are super friendly and the menu is good .
Cross-AE	we went on the car of the time and the chicken was awful .	Cross-AE	the place , one of the office is always worth you do a business .
Original	service is good but not quick , just enjoy the wine and your company .	Original	they told us in the beginning to make sure they do n't eat anything .
ARAE	service is good but not quick , but the service is horrible .	ARAE	they told us in the mood to make sure they do great food .
Cross-AE	service is good , and horrible , is the same and worst time ever .	Cross-AE	they 're us in the next for us as you do n't eat .
Original	the steak was really juicy with my side of salsa to balance the flavor .	Original	the person who was teaching me how to control my horse was pretty rude .
ARAE	the steak was really bland with the sauce and mashed potatoes .	ARAE	the person who was able to give me a pretty good price .
Cross-AE	the fish was so much , the most of sauce had got the flavor .	Cross-AE	the owner 's was gorgeous when i had a table and was friendly .
Original	other than that one hell hole of a star bucks they 're all great !	Original	he was cleaning the table next to us with gloves on and a rag .
ARAE	other than that one star rating the toilet they 're not allowed .	ARAE	he was prompt and patient with us and the staff is awesome .
Cross-AE	a wonder our one came in a _num_ months , you 're so better !	Cross-AE	he was like the only thing to get some with with my hair .

Table 4: Full sheet of sentiment transfer result

Yahoo Transfer

	from Science		from Music		from Politics
Original	what is an event horizon with regards to black holes ?	Original	do you know a website that you can find people who want to join bands ?	Original	republicans : would you vote for a cheney / satan ticket in 2008 ?
Music	what is your favorite sitcom with adam sandler ?	Science	do you know a website that can help me with science ?	Science	guys : how would you solve this question ?
Politics	what is an event with black people ?	Politics	do you think that you can find a person who is in prison ?	Music	guys : would you rather be a good movie ?
Original	what did john paul jones do in the american revolution ?	Original	do people who quote entire poems or song lyrics ever actually get chosen best answer ?	Original	if i move to the usa do i lose my pension in canada ?
Music	what did john lennon do in the new york family ?	Science	do you think that scientists learn about human anatomy and physiology of life ?	Science	if i move the <unk> in the air i have to do my math homework ?
Politics	what did john mccain do in the next election ?	Politics	do people who knows anything about the recent issue of <unk> leadership ?	Music	if i move to the music do you think i feel better ?
Original	can anybody suggest a good topic for a statistical survey ?	Original	from big brother , what is the girls name who had <unk> in her apt ?	Original	what is your reflection on what will be our organizations in the future ?
Music	can anybody suggest a good site for a techno ?	Science	in big bang what is the <unk> of <unk> , what is the difference between <unk> and <unk> ?	Science	what is your opinion on what will be the future in our future ?
Politics	can anybody suggest a good topic for a student visa ?	Politics	is big brother in the <unk> what do you think of her ?	Music	what is your favorite music videos on the may i find ?
Original	can a kidney infection effect a woman 's <unk> cycle ?	Original	where is the tickets for the filming of the suite life of zack and cody ?	Original	wouldn 't it be fun if we the people veto or passed bills ?
Music	can anyone give me a good film <unk> ?	Science	where is the best place of the blood stream for the production of the cell ?	Science	isnt it possible to be cloned if we put the moon or it ?
Politics	can a landlord officer have a <unk> <unk> ?	Politics	where is the best place of the navy and the senate of the union ?	Music	isnt it possible or if we 're getting married ?
Original	where does the term " sweating <unk> " come from ?	Original	the <unk> singers was a band in 1963 who had a hit called <unk> man ?	Original	can anyone tell me how i could go about interviewing north vietnamese soldiers ?
Music	where does the term " <unk> " come from ?	Science	the <unk> river in a <unk> was created by a <unk> who was born in the last century ?	Science	can anyone tell me how i could find how to build a robot ?
Politics	where does the term " <unk> " come from ?	Politics	the <unk> are <unk> in a <unk> who was shot an <unk> ?	Music	can anyone tell me how i could find out about my parents ?
Original	what other <unk> sources are there than burning fossil fuels .	Original	what is the first metal band in the early 60 's ..... ? ? ? ?	Original	if the us did not exist would the world be a better place ?
Music	what other <unk> are / who are the greatest guitarist currently on tv today ?	Science	what is the first country in the universe ?	Science	if the world did not exist , would it be possible ?
Politics	what other <unk> are there for veterans who lives ?	Politics	who is the first president in the usa ?	Music	if you could not have a thing who would it be ?

Table 5: Full sheet of Yahoo titles transfer result

	from Science		from Music		from Politics
Original	take 1ml of hcl (concentrated) and dilute it to 50ml .	Original	all three are fabulous artists , with just incredible talent ! !	Original	4 years of an idiot in office + electing the idiot again = ?
Music	take em to you and shout it to me	Science	all three are genetically bonded with water , but just as many substances , are capable of producing a special case .	Science	4 years of an idiot in the office of science ?
Politics	take bribes to islam and it will be punished .	Politics	all three are competing with the government , just as far as i can .	Music	4) <unk> in an idiot , the idiot is the best of the two points ever !
Original	oils do not do this , they do not " set " .	Original	she , too , wondered about the underwear outside the clothes .	Original	send me $ 100 and i 'll send you a copy - honest .
Music	cucumbers do not do this , they do not " do " .	Science	she , too , i know , the clothes outside the clothes .	Science	send me an email and i 'll send you a copy .
Politics	corporations do not do this , but they do not .	Politics	she , too , i think that the cops are the only thing about the outside of the u.s. .	Music	send me $ 100 and i 'll send you a copy .
Original	the average high temps in jan and feb are about 48 deg .	Original	i like rammstein and i don 't speak or understand german .	Original	wills can be <unk> , or typed and signed without needing an attorney .
Music	the average high school in seattle and is about 15 minutes .	Science	i like googling and i don 't understand or speak .	Science	euler can be <unk> , and without any type of operations , or <unk> .
Politics	the average high infantry division is in afghanistan and alaska .	Politics	i like mccain and i don 't care about it .	Music	madonna can be <unk> , and signed without opening or <unk> .
Original	the light from you lamps would move away from you at light speed	Original	mark is great , but the guest hosts were cool too !	Original	hungary : 20 january 1945 , (formerly a member of the axis)
Music	the light from you tube would move away from you	Science	mark is great , but the water will be too busy for the same reason .	Science	nh3 : 20 january , 78 (a)
Politics	the light from you could go away from your state	Politics	mark twain , but the great lakes , the united states of america is too busy .	Music	1966 - 20 january 1961 (a) 1983 song
Original	van <unk> , on the other hand , had some serious issues ...	Original	they all offer terrific information about the cast and characters , ...	Original	bulgaria : 8 september 1944 , (formerly a member of the axis)
Music	van <unk> on the other hand , had some serious issues .	Science	they all offer insight about the characteristics of the earth , and are composed of many stars .	Science	moreover , 8 $\hat{3}$ + (x + 7) (x $\hat{2}$) = (a $\hat{2}$)
Politics	van <unk> , on the other hand , had some serious issues .	Politics	they all offer legitimate information about the invasion of iraq and the u.s. , and all aspects of history .	Music	harrison : 8 september 1961 (a) (1995)
Original	just multiply the numerator of one fraction by that of the other .	Original	but there are so many more i can 't think of !	Original	anyone who doesnt have a billion dollars for all the publicity cant win .
Music	just multiply the fraction of the other one that 's just like it .	Science	but there are so many more of the number of questions .	Science	anyone who doesnt have a decent chance is the same for all the other .
Politics	just multiply the same fraction of other countries .	Politics	but there are so many more of the can i think of today .	Music	anyone who doesnt have a lot of the show for the publicity .
Original	civil engineering is still an umbrella field comprised of many related specialties .	Original	i love zach he is sooo sweet in his own way !	Original	the theory is that cats don 't take to being tied up but thats <unk> .
Music	civil rights is still an art union .	Science	the answer is he 's definitely in his own way !	Science	the theory is that cats don 't grow up to <unk> .
Politics	civil law is still an issue .	Politics	i love letting he is sooo smart in his own way !	Music	the theory is that dumb but don 't play <unk> to <unk> .
Original	h2o2 (hydrogen peroxide) naturally decomposes to form o2 and water .	Original	remember the industry is very shady so keep your eyes open !	Original	the fear they are trying to instill in the common man is based on what ?
Music	jackie and brad pitt both great albums and they are my fav .	Science	remember the amount of water is so very important .	Science	the fear they are trying to find the common ancestor in the world .
Politics	kennedy and blair hate america to invade them .	Politics	remember the amount of time the politicians are open your mind .	Music	the fear they are trying to find out what is wrong in the song .
Original	the quieter it gets , the more white noise you can here .	Original	but can you fake it , for just one more show ?	Original	think about how much planning and people would have to be involved in what happened .
Music	the fray it gets , the more you can hear .	Science	but can you fake it , just for more than one ?	Science	think about how much time would you have to do .
Politics	the gop gets it , the more you can here .	Politics	but can you fake it for more than one ?	Music	think about how much money and what would be <unk> about in the world ?
Original	h2co3 (carbonic acid) naturally decomposes to form water and co2 .	Original	i am going to introduce you to the internet movie database .	Original	this restricts the availability of cash to them and other countries too start banning them .
Music	phoebe and jack , he 's gorgeous and she loves to get him !	Science	i am going to investigate the internet to google .	Science	this reduces the intake of the other molecules to produce them and thus are too large .
Politics	nixon (captured) he lied and voted for bush to cause his country .	Politics	i am going to skip the internet to get you checked .	Music	this is the cheapest package of them too .

Table 6: Full sheet of Yahoo answers transfer result

Natural Language Multitasking

Analyzing and Improving Syntactic Saliency of Hidden Representations

Gino Brunner, Yuyi Wang, Roger Wattenhofer, Michael Weigelt*
ETH Zurich, Switzerland
{brunnegi,yuwang,wattenhofer,weigeltm}@ethz.ch

Abstract

We train multi-task autoencoders on linguistic tasks and analyze the learned hidden sentence representations. The representations change significantly when translation and part-of-speech decoders are added. The more decoders a model employs, the better it clusters sentences according to their syntactic similarity, as the representation space becomes less entangled. We explore the structure of the representation space by interpolating between sentences, which yields interesting pseudo-English sentences, many of which have recognizable syntactic structure. Lastly, we point out an interesting property of our models: The difference-vector between two sentences can be added to change a third sentence with similar features in a meaningful way.

1 Introduction

Representation learning has opened the doors for many creative neural networks that learn to generate music or extract the artistic style of a painting and apply it to an arbitrary photograph (Gatys et al. [2016]). In computational linguistics, progress has been made in neural machine translation, speech-to-text and many other applications. However, creative algorithms that write poetry, mimic an author or even develop fictional languages are sparse or non-existent. Since good representations are crucial for such creative tasks, we examine ways to improve learned representations and develop ways to measure their linguistic quality. We analyze how improvements in the syntactic capabilities of a model relate to the learned hidden representations. A syntax clustering experiment shows that the representation space of multi-task models is more easily separable into disentangled regions than that of single-task models. As a result, some simple sentence features can be added and subtracted from each other in the representation space. Our work does not focus on optimizing one single task or error, but on forcing the representations to contain useful information in a structured, analyzable way.

To this end we train several sequence to sequence models with an increasing number of decoders. Each decoder has a distinctive linguistic task. We compare the sentence representations our models have learned and explore how representations of different sentences relate to each other.[2]

2 Related work

Sutskever et al. [2014] use Long Short-Term Memory (LSTM) networks to encode an arbitrary sequence into a vector and decode it back into a (possibly different) sequence. They achieve results in neural machine translation (NMT) that are competitive with statistical machine translation models (SMT). The NMT objective is one of several tasks we use in our models. Luong et al. [2015] extend Sutskever et al.'s model with three multi-task settings: One-to-many, many-to-one and many-to-many.

*Authors are listed in alphabetical order.

[2]Our code is available at https://drive.google.com/open?id=0B7Mps2rt3vBoSHlWeElGT1JiS3c

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Their work shows that translation performance can benefit from parsing and image caption tasks. This kind of improvement from adding related tasks is also the subject of our work. Niehues and Cho [2017] note that many linguistic resources that enabled SMT are not commonly used in NMT models. They train translation models jointly with part-of-speech (POS) and named-entity recognition tasks and show that both translation and POS tagging benefit from the shared information. Our research is rooted in the same idea, but we focus on the learned representations rather than on training objectives.

As an alternative to the bag-of-words feature many natural language processing (NLP) models use, Le and Mikolov [2014] propose the "paragraph vector" to represent sentences, paragraphs and whole documents. This feature outperforms bag-of-word models in text classification and sentiment analysis tasks. While it could be extended to deal with larger pieces of text, our work focuses on the sentence-level. Liu et al. [2015] developed a multi-task deep neural network for multi-domain classification and information retrieval, which learns general semantic representations useful for both tasks, demonstrating the advantage of multi-task learning. Artetxe et al. [2017] note that large parallel corpora for the training of NMT models are scarce. They introduce a novel system that solely relies on monolingual data while still learning to translate between languages. Vinyals et al. [2015] develop a generative model that connects image processing and natural language generation. Their model takes an image as input and generates an English sentence that describes the content of the image. Such a complex task relies on good, well-generalized representations, which we are exploring in this work as well.

3 Models

The model architecture we use to learn representations is the autoencoder, which operates in two stages: An encoder transforms data into a "code" in a hidden layer, from which the decoder then tries to reconstruct the original input. The decoder can be modified to learn a task other than reconstruction, such as translating the input into a different language. Because text is of sequential nature we use Long Short Term Memory recurrent networks (LSTM) as encoders and decoders.

3.1 Multi-task autoencoder

We extend the basic sequence to sequence autoencoder model by adding multiple decoders that perform separate linguistic tasks. First, an encoder LSTM consumes a sequence of characters one by one and updates its internal state. When the whole input sequence has been read, the encoder state contains information about the entire sequence. This state is fed into a dense layer which we call the "representation layer", the output of which is a real vector with a specified dimension. The analysis we perform in Section 4 refers to the output of this layer. Next, the representation vector is fed to each decoder LSTM, which then generate output sequences corresponding to their tasks. A single-task model will learn representations of its training data which are useful for the objective at hand. Since it supports multiple decoders, our model architecture forces the representations to contain useful information for each objective. Adding more linguistic tasks as decoders should make representations more salient from a linguistic perspective and change the properties of the whole representation space in a meaningful and analyzable way.

3.2 Decoders

We use four different decoders. The replicating (REP) decoder's task is to reconstruct the input sequence. The German and French (DE/FR) decoders attempt to translate the input sentence to German and French respectively. The last decoder we use learns to tag words in the input sequence with part-of-speech tags (POS), such as *verb*, *noun*, *adjective*. Figure 1 shows the architecture of our multi-task autoencoder model.

3.3 Dataset

To train our models on the three tasks replication, translation and part-of-speech-tagging, we require a multilingual corpus with sentences that correspond to each other. The transcripts of the European Parliament sessions (Koehn [2005]) are a suitable corpus with aligned English, German and French sentences. The replication task uses the English sentences as both input and target. The training data for the POS decoders was created using the python nltk module (Loper and Bird [2002]) and

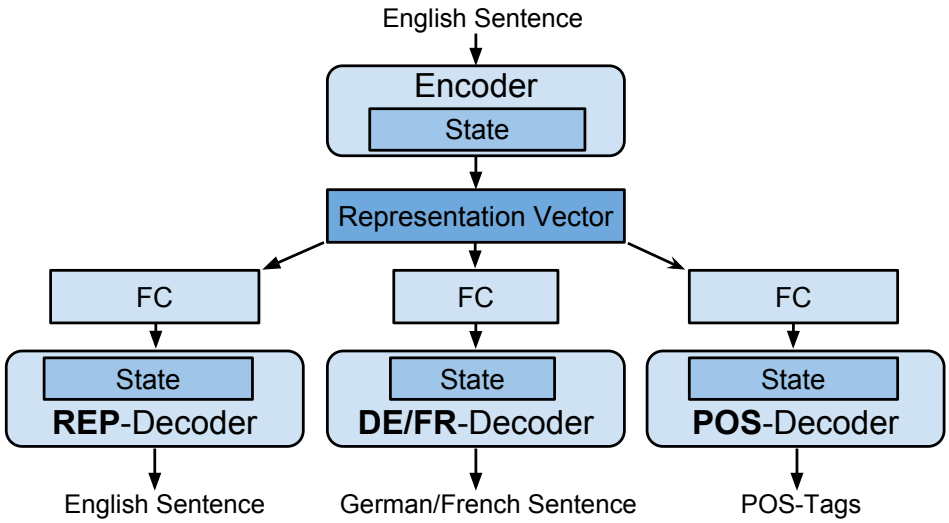


Figure 1: Architecture of our multi-task autoencoder models. We use four different decoders: Replicating (REP), translation to German (DE), translation to French (FR) and a part-of-speech tagger (POS). The encoder and decoders are LSTMs. The fully connected (FC) layers transform the fixed-length representation vectors into variable length state vectors for each decoder.

the English text from the European Parliament corpus. The subset of this dataset we use contains over 1.7 million sentences (for all decoders), 1.5 million of which were used as the training set. The remaining 0.2 million sentences form the test set. A training example is a tuple whose size depends on the model configuration. For example, the single-task REP model uses 2-tuples (input: English sentence, target: English sentence), whereas the multi-task REP-DE-POS-model uses 4-tuples (input: English sentence, REP-target: English sentence, DE-target: German sentence, POS-target: POS-tag sequence). While the number of available training examples is the same for each model, multi-task models are trained with larger tuples and therefore more training data. To account for this imbalance, we train reference models with fewer training examples and compare their performance to the main models which we train on the full dataset.

3.4 Model configurations

We train models with different decoders and representation layer sizes. Table 1 shows the perplexities reached by each decoder for some of the trained models. Perplexity measures how close a generated sequence is to a target sequence and is defined as the exponential of the cross entropy between the two sequences. The model name simply lists its decoders and representation layer size. All encoder and decoder LSTMs have 512 neurons. The achieved perplexities are not competitive with state-of-the-art models. However, our work focuses on learned representations, not on decoder losses. The perplexities reached by the reference models mentioned in 3.3 are indistinguishable from the main models, and are thus not listed separately.

Table 1: Model Configurations

Model name	REP	DE	FR	POS
REP-1024	1.05	-	-	-
REP-DE-1024	1.04	2.02	-	-
REP-DE-FR-1024	1.04	2.02	1.86	-
REP-DE-256	1.07	2.02	-	-
REP-DE-POS-256	1.14	2.05	-	1.10

4 Results

4.1 Syntax clustering

How can the quality of learned representations be measured?

The goal we pursue with our models is not the highest possible decoder accuracy. Instead, we are interested in representations that capture some linguistic aspect of the input language. In order to compare the learned representations of different models, we examine how well they cluster syntactically similar sentences. We use 14 sentence prototypes with different syntactic structures, for example "The +N is +A", where +N, +V, +A and +D are placeholders for nouns, verbs, adjectives and adverbs. Following is a list of all 13 sentence prototypes we used (the 14th category are the empty sentences):

The +N is +A.
The +N +Vs.
The +N has a +N.
The +N +Vs a +N.
The +N +Vs +D.
No +N ever +Vs.
Are +Ns +A?
The +Ns of +N +D +V the +A +N, but some +Ns still +V their +N.
In the +N of a +A +N, the +N will +V the +N of +Ving the +N.
+Ns +V the +A +N of +Ns +Ving on the +N.
In the +N of +N, +Ns would rather +V without +N than +V any +A +Ns.
+N +Vs in order to +V on a +N.
+A +Ns often +V like +Ns.

Each sentence prototype is randomly populated by common English words 100 times. The syntax of each sentence in such a category is very similar or identical to all others in the same category, but different from sentences in other categories. These sentences are then fed into our models. We record every resulting representation and pair it with its input sentence. Using K-means clustering with $K = 14$ we cluster the representation-sentence pairs in the representation space. For each resulting cluster, we count how many sentences of each prototype it contains. This yields a list such as this: $[30, 3, 1, 7, 88, 0, 0, 0, 0, 0, 0, 0, 0, 0]$, which shows the content of one of 14 clusters: 30 sentences of type one, 3 of type two and so on. Since most sentences in this cluster are of type five, this cluster is assigned to be the cluster of sentence category five. However, 41 sentences of type other than five were "falsely" assigned to this cluster. Therefore, the error of this cluster is 41. The sum of errors of all 14 clusters is the *clustering error*, which is our quality measure for this experiment. Since K-means clustering is nondeterministic, we run the algorithm 100 times. Table 2 shows the best-of-100 clustering errors.

Table 2: Clustering errors by model

Model	R-1024	R-D-1024	R-D-F-1024	R-D-256	R-D-P-256
Best	149	22	24	29	3

Starting from the left, the single-task REP model has the highest clustering error. The second model clusters significantly better because we added a German translation decoder. The third model adds a French translation decoder on top of the German one, but that does not improve the clusterings. Still, the difference between a single REP and the REP-DE combination supports our hypothesis that the quality of language representations improve with additional linguistic tasks.

The clustering error of the R-D-256 is not significantly different from the R-D-1024 model, which indicates that the representation size can be reduced by a large factor while neither affecting this particular measure of syntactic quality nor the decoder loss.

The clustering error of the rightmost model, which was trained with a POS decoder, is almost zero. This means that the 14 syntactically different sentence prototypes are well separated in the representation space. It is not surprising that this model performs better than the others: At least for humans and most classical algorithms, correct POS tagging is a requirement or preparative step to syntax analysis. The distinctive advantage this model has over the others indicates that neural

language models benefit from related linguistic tasks. Having clear, separable clusters of sentences suggests that some aspects of syntax are disentangled in the multi-task representation space.

We trained several reference models on fewer training examples to account for the different numbers of decoders, and thus different amount of effective training data (as described in Section 3.3). Although the clustering errors differ, there is no clear trend, as some models cluster worse and others better with fewer examples (see Table 3). Note that the perplexities these models achieved are all comparable to their reference models (trained with the full training set), and none of the models overfit the training data.

Table 3: Best-of-100 clustering errors with fewer training examples

Model	R-D-1024	R-D-256	R-D-F-1024	R-D-P-256
Full training set	22	29	24	3
1/2 training set	37	33	-	-
1/3 training set	-	-	19	1

4.2 Interpolation

The sentence representations our models generate lie in a high-dimensional space. Clustering experiments show that the data points from sentences in the training set are not spaced evenly in this vector space, rather they form clusters or manifolds. Seeing how clearly some models cluster sentences according to their syntax, the question arises: What lies between two sentences? More precisely: If two sentences s_1 and s_2 have representations r_1 and r_2, what sentence corresponds to $\frac{r_1+r_2}{2}$? What about other points along a straight line between r_1 and r_2? Table 4 shows two example outputs each of the REP decoder of the REP-DE-POS-256 and REP-DE-1024 models. As shown in Section 4.1, the REP-DE-POS-256 has a significantly lower clustering error, and it can be seen that it also produces more plausible sentences with fewer non-words when doing interpolation in the representation space.

Table 4: Sentence interpolation between two endpoints for two different models.

REP-DE-POS-256:	REP-DE-1024:
Is this what we want?	Is this what we want?
Is this meat which we need?	Is this what nath we affend?
Is that is very with true?	Whin to shakin the weaknes fan?
Whe nomists ha told day of items.	Why cust hesitage we chear thembox.
The course all hotels to drug itselve.	The consumer is of encautant where quote.
The consumer may took speed in follows Mr...	The consumer is botted there binds for EU.
The consumer must therefore be informed of GMOs.	The consumer must therefore be informed of GMOs
We will not tolerate a policy of religious repression.	We will not tolerate a policy of religious repression.
We will not threaten by insist regarding our representation.	We will not dossil at Ecoprat south direct-resprest and.
We must the knowled alro in economic objarimar represent.	We sant techni oy asplig poor correads in report'sed.
Whe is still want recoursion in viruccuroning responsibly.	The interest has lip porals often Mrs Martensrespre.
The issue where any returning ideology rrrengeing reply.	The insade to sarame places outso in requirponts resel.
There is a systemic policy of religious reprovement.	There is a strongly matic poorer 'fragen presumers' thre.
There is a systematic policy of religious repression.	There is a systematic policy of religious repression.

Unsurprisingly, not every point in the representation space corresponds to a correct English sentence. If we overlook the non-words in some of the interpolated sentences, the change in syntax can be understood to a degree. However, linear interpolation does not seem to be enough to explore the shape of this representation space. More work investigating these manifolds could yield useful results for generative, creative algorithms. For example, what properties would a representation space have if it were trained to understand two languages? How would representations of sentences and words from different languages relate to each other?

4.3 Representation "arithmetic"

Some word embedding spaces have a special property, where differences between embedded words can correspond to a direction on an axis that has a semantic or grammatical interpretation. For example, the difference vector of *queen* and *king* might roughly equal that of *woman* and *man* (Mikolov et al. [2013]). This raises the question whether sentence representations can have similar properties. A simple assumption would be that the difference-vector of *Cats are good pets.* and *Dogs are good pets.* should have canceled out the part about good pets and roughly point from *Dogs* to *Cats*. Adding this difference-vector to any sentence that contains *Dogs* should then result in a sentence where *Dogs* is replaced with *Cats*. The REP-DE-1024 model fails at this task, as shown in Table 5. However, the arithmetic works better for the model that performed best in the syntax clustering (REP-DE-POS-256) experiment, as is shown in Table 6. The arithmetic worked for first three sentences, and failed for the last two. While these results are not representative and require more rigorous investigation, we did not cherry pick the examples.

The fact that the representation "arithmetic" works for a number of small examples shows that the learned representations are much more complex than mere symbol probabilities. Since this experiment only considers word occurrence and order, it remains open whether there is any semantic component to this phenomenon. Adding specialized semantic tasks to our models could improve the results.

Table 5: Representation "arithmetic" for the REP-DE-1024 model.

s_1		s_2		s_3		s_1 - s_2 + s_3
1: I am one.	-	I am two.	+	You are two.	=	You ready no.
2: This example works.	-	This example fails.	+	Another attempt fails.	=	Another attempt world.
3: A word in a phrase.	-	A tree in a phrase.	+	A tree is green.	=	A word is purevy?
4: The end is easier.	-	The start is easier.	+	A start is next!	=	A rew lieh in new!
5: A large number of people want to work.	-	A small number of people want to work.	+	A small sentence is enough.	=	A large senselfeir in or evacce.

Table 6: Representation "arithmetic" for the REP-DE-POS-256 model.

s_1		s_2		s_3		s_1 - s_2 + s_3
1: I am one.	-	I am two.	+	You are two.	=	You are one.
2: This example works.	-	This example fails.	+	Another attempt fails.	=	Another attempts work.
3: A word in a phrase.	-	A tree in a phrase.	+	A tree is green.	=	A word is green.
4: The end is easier.	-	The start is easier.	+	A start is next!	=	A need aid not!
5: A large number of people want to work.	-	A small number of people want to work.	+	A small sentence is enough.	=	A large sector for challenge.

5 Conclusion and Future Work

We trained several multi-task autoencoders on linguistic tasks and analyzed the learned sentence representations. The representations change significantly when translation and part-of-speech tagging decoders are added. The more decoders a model uses, the better it can cluster sentence representations according to their syntactic similarity. This indicates that the space (at least the part of it that is associated with syntactic information) becomes more separable or disentangled as more tasks are added.

We explored the structure of the representation space by interpolating between sentences, which yields interesting pseudo-English sentences, many of which have recognizable syntactic structure. Finally, we point out an interesting property of our models' representations: The difference-vector between two sentence representations can be added to change a third sentence with similar features in a meaningful way. We call this process "representation arithmetic", since it allows adding and subtracting sentence features to and from other sentences.

In the future, we want to get a better understanding of the shape of the representation space. Interpolating inside the manifold the data populates could enable creative algorithms which produce grammatical sentences by sampling from the inside of the manifold. Perhaps the "representation arithmetic" property can be made more robust by adding semantic tasks as decoders. If this behavior

could be made more predictable, the representation space would have useful properties for generative models, as semantic features could be transferred between sentences.

References

Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pages 2414–2423, 2016.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems, NIPS 2014*, pages 3104–3112, 2014.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *CoRR*, abs/1511.06114, 2015.

Jan Niehues and Eunah Cho. Exploiting linguistic resources for neural machine translation using multi-task learning. In *Proceedings of the Second Conference on Machine Translation, WMT 2017*, pages 80–89, 2017.

Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014*, pages 1188–1196, 2014.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 912–921, 2015.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. *CoRR*, abs/1710.11041, 2017.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3156–3164, 2015.

Philipp Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. 2005.

Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70. Association for Computational Linguistics, 2002.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Annual Conference on Neural Information Processing Systems, NIPS 2013*, pages 3111–3119, 2013.

Variational Inference of Disentangled Latents from Unlabeled Observations

Abhishek Kumar
IBM Research AI
Yorktown Heights, NY
abhishk@us.ibm.com

Prasanna Sattageri
IBM Research AI
Yorktown Heights, NY
psattig@us.ibm.com

Avinash Balakrishnan
IBM Research AI
Yorktown Heights, NY
avinash.bala@us.ibm.com

Abstract

Disentangled representations, where the higher level data generative factors are reflected in disjoint latent dimensions, offer several benefits such as ease of deriving invariant representations, transferability to other tasks, interpretability, etc. We consider the problem of unsupervised learning of disentangled representations from large pool of unlabeled observations, and propose a variational inference based approach to infer disentangled latent factors. We introduce a regularizer on the expectation of the approximate posterior over observed data that encourages the disentanglement. We evaluate the proposed approach using several quantitative metrics and empirically observe significant gains over existing methods in terms of both disentanglement and data likelihood (reconstruction quality).

1 Introduction

Feature representations of the observed raw data play a crucial role in the success of machine learning algorithms. Effective representations should be able to capture the underlying (abstract or high-level) latent generative factors that are relevant for the end task while ignoring the inconsequential or nuisance factors. *Disentangled* feature representations have the property that the generative factors are revealed in disjoint subsets of the feature dimensions, such that a change in a single generative factor causes a highly sparse change in the representation. Disentangled representations offer several advantages in terms of deriving invariant representations, transferability to various tasks, interpretability, etc. Indeed, the importance of learning disentangled representations has been argued in several recent works [1, 12, 18].

Recognizing the significance of disentangled representations, several attempts have been made in this direction in the past [18], however much of the earlier work assumes some sort of supervision [16, 21, 11, 8]. However, in most real scenarios, we only have access to raw observations without any supervision about the generative factors. Recently, Chen et al. [3] proposed an approach to learn a generative model with disentangled factors based on Generative Adversarial Networks (GAN) [6], however implicit generative models like GANs lack an effective inference mechanism, which hinders its applicability to the problem of inferring disentangled representations. More recently, Higgins et al. [7] proposed an approach based on Variational AutoEncoder (VAE) [10] for inferring disentangled factors. The inferred latents using their method (termed as β-VAE) are empirically shown to have better disentangling properties, however the method deviates from the basic principles of variational inference, creating increased tension between observed data likelihood and disentanglement. This in turn leads to poor quality of the generated samples as observed in [7].

In this work, we propose a principled approach for inference of disentangled latent factors based on the popular and scalable framework of Variational Autoencoders [10, 19, 5, 17]. Disentanglement is encouraged by introducing a regularizer over the induced *inferred prior*. Unlike β-VAE [7], our approach does not introduce any extra conflict between disentanglement of the latents and the observed data likelihood, which is reflected in the overall quality of the generated samples that matches the VAE and is much better than β-VAE. This does *not* come at the cost of higher

entanglement and our approach also outperforms β-VAE in disentangling the latents as measured by the quantitative metrics.

2 Formulation

We start with a generative model of the observed data that first samples a latent variable $\mathbf{z} \sim p(\mathbf{z})$, and an observation is generated by sampling from $p_\theta(\mathbf{x}|\mathbf{z})$. The joint density of latents and observations is denoted as $p_\theta(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p_\theta(\mathbf{x}|\mathbf{z})$. The problem of inference is to compute the posterior of the latents conditioned on the observations, i.e., $p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_\theta(\mathbf{x},\mathbf{z})}{\int p_\theta(\mathbf{x},\mathbf{z})d\mathbf{z}}$. We assume that we are given a finite set of samples (observations) from the true data distribution $p(\mathbf{x})$. In most practical scenarios involving high dimensional and complex data, this computation is intractable and calls for approximate inference. Variational inference takes an optimization based approach to this, positing a family $\mathcal{D}$ of approximate densities over the latents and reducing the approximate inference problem to finding a member density that minimizes the Kullback-Leibler divergence to the true posterior, i.e., $q^*_{\mathbf{x}} = \min_{q\in\mathcal{D}} \mathrm{KL}(q(\mathbf{z})\|p_\theta(\mathbf{z}|\mathbf{x}))$ [2]. The idea of amortized inference [10, 19, 5, 17] is to explicitly share information across inferences made for each observation. One successful way of achieving this for variational inference is to have a so-called *recognition model*, parameterized by ϕ, that encodes an inverse map from the observations to the approximate posteriors (also referred as variational autoencoder or VAE) [10, 17]. The recognition model parameters are learned by optimizing the problem $\min_\phi \mathbb{E}_{\mathbf{x}}\mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x}))$, where the outer expectation is over the true data distribution $p(\mathbf{x})$ which we have samples from. This can be shown as equivalent to maximizing what is termed as evidence lower bound (ELBO):

$$\arg\min_{\theta,\phi} \mathbb{E}_{\mathbf{x}}\mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})) = \arg\max_{\theta,\phi} \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))\right] \quad (1)$$

The ELBO (the objective at the right side of Eq. 1) lower bounds the log-likelihood of observed data, and the gap vanishes at the global optimum. Often, the density forms of $p(\mathbf{z})$ and $q_\phi(\mathbf{z}|\mathbf{x})$ are chosen such that their KL-divergence can be written analytically in a closed-form expression (e.g., $p(\mathbf{z})$ is $N(0, I)$ and $q_\phi(\mathbf{z}|\mathbf{x})$ is $N(\mu_\phi(\mathbf{x}), \Sigma_\phi(\mathbf{x}))$) [10]. In such cases, the ELBO can be efficiently optimized (to a stationary point) using stochastic first order methods where both expectations are estimated using mini-batches. Further, in cases when $q_\phi(\cdot)$ can be written as a continuous transformation of a fixed base distribution (e.g., the standard normal distribution), a low variance estimate of the gradient over ϕ can be obtained by coordinate transformation (also referred as reparametrization) [4, 10, 17].

2.1 Generative story: disentangled prior

Most VAE based generative models for real datasets (e.g., text, images, etc.) already work with a relatively simple and disentangled prior $p(\mathbf{z})$ having no interaction among the latent dimensions (e.g., the standard Gaussian $N(0, I)$) The complexity of the observed data is absorbed in the conditional distribution $p_\theta(\mathbf{x}|\mathbf{z})$ which encodes the interactions among the latents. Hence, as far as the generative modeling is concerned, disentangled prior sets us in the right direction.

2.2 Inferring disentangled latents

Although the generative model starts with a disentangled prior, our main objective is to *infer* disentangled latents which are potentially conducive for various goals mentioned in Sec. 1 (e.g., invariance, transferability, interpretability). To this end, we consider the density over the inferred latents induced by the approximate posterior inference mechanism, $q_\phi(\mathbf{z}) = \int q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x})d\mathbf{x}$, which we will subsequently refer to as the *inferred prior* or *expected variational posterior* ($p(\mathbf{x})$ is the true data distribution that we have only samples from). For inferring disentangled factors, this should be factorizable along the dimensions, i.e., $q_\phi(\mathbf{z}) = \prod_i q_i(z_i)$, or equivalently $q_{i|j}(z_i|z_j) = q_i(z_i)$, $\forall\, i, j$. This can be achieved by minimizing a suitable distance between the inferred prior $q_\phi(\mathbf{z})$ and the disentangled generative prior $p(\mathbf{z})$. We can also define *expected posterior* as $p_\theta(\mathbf{z}) = \int p_\theta(\mathbf{z}|\mathbf{x})p(\mathbf{x})d\mathbf{x}$. If we take KL-divergence as our choice of distance, by relying on its pairwise convexity (i.e., $\mathrm{KL}(\lambda p_1 + (1-\lambda)p_2\|\lambda q_1 + (1-\lambda)q_2) \leq \lambda\mathrm{KL}(p_1\|q_1) + (1-\lambda)\mathrm{KL}(p_2\|q_2)$) [20], we can show that the distance between $q_\phi(\mathbf{z})$ and $p_\theta(\mathbf{z})$ is bounded by the objective of the variational inference (the ELBO in Eq. (1)):

$$\mathrm{KL}(q_\phi(\mathbf{z})\|p_\theta(\mathbf{z})) = \mathrm{KL}(\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}q_\phi(\mathbf{z}|\mathbf{x})\|\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}p_\theta(\mathbf{z}|\mathbf{x})) \leq \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\,\mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})). \quad (2)$$

In general, the prior $p(\mathbf{z})$ and expected posterior $p_\theta(\mathbf{z})$ will be different, although they may be close (they will be same when $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$ is equal to $p(\mathbf{x})$). Hence, variational posterior

inference of latent variables with disentangled prior naturally encourages inferring factors that are *close* to being disentangled. We think this is the reason that the original VAE (Eq. (1) has also been observed to exhibit some disentangling behavior on simple datasets such as MNIST [10]. However, this behavior does not carry over to more complex datasets [7], unless extra supervision on the generative factors is provided [11, 9]. This can be due to: (i) $p(\mathbf{x})$ and $p_\theta(\mathbf{x})$ being far apart which in turn causes $p(\mathbf{z})$ and $p_\theta(\mathbf{z})$ being far apart, and (ii) the non-convexity of the ELBO objective which prevents us from achieving the global minimum of $\mathbb{E}_{\mathbf{x}}\mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x}))$ (which is 0 and implies $\mathrm{KL}(q_\phi(\mathbf{z})\|p_\theta(\mathbf{z})) = 0$). In other words, maximizing the ELBO (Eq. (1)) might also result in reducing the value of $\mathrm{KL}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$, however, due to the aforementioned reasons, the gap between $\mathrm{KL}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$ and $\mathbb{E}_{\mathbf{x}}\,\mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x}))$ could be large at the stationary point of convergence. Hence, minimizing $\mathrm{KL}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$ explicitly will give us better control on the disentanglement. This motivates us to add $\mathrm{KL}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$ as part of the objective to encourage disentanglement during inference, i.e.,

$$\max_{\theta,\phi} \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))\right] - \lambda\,\mathrm{KL}(q_\phi(\mathbf{z})\|p(\mathbf{z})), \tag{3}$$

where λ controls its contribution to the overall objective.

Optimizing (3) directly is not tractable due to the presence of $\mathrm{KL}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$ which does not have a closed-form expression. One possibility is use the variational formulation of the KL-divergence [14, 15] that needs only samples from $q_\phi(\mathbf{z})$ and $p(\mathbf{z})$ to estimate a lower bound to $\mathrm{KL}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$. However, this would involve optimizing for a third set of parameters ψ for the KL-divergence estimator, and would also change the optimization to a saddle-point (min-max) problem which has its own challenges. We adopt a simpler yet effective alternative of matching the moments of the two distributions. In particular, we match the covariance of the two distributions which will amount to decorrelating the dimensions of $\mathbf{z} \sim q_\phi(\mathbf{z})$ if $p(\mathbf{z})$ is $N(0, I)$. Let us denote $\mathrm{Cov}_{q(\mathbf{z})}(\mathbf{z}) := \mathbb{E}_{q(\mathbf{z})}\left[(\mathbf{z} - \mathbb{E}_{q(\mathbf{z})}(\mathbf{z}))(\mathbf{z} - \mathbb{E}_{q(\mathbf{z})}(\mathbf{z}))^\top\right]$. By the law of total covariance, the covariance of $\mathbf{z} \sim q_\phi(\mathbf{z})$ is given by

$$\mathrm{Cov}_{q_\phi(\mathbf{z})}(\mathbf{z}) = \mathbb{E}_{p(\mathbf{x})}\mathrm{Cov}_{q_\phi(\mathbf{z}|\mathbf{x})}(\mathbf{z}) + \mathrm{Cov}_{p(\mathbf{x})}\left(\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}(\mathbf{z})\right), \tag{4}$$

where $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}(\mathbf{z})$ and $\mathrm{Cov}_{q_\phi(\mathbf{z}|\mathbf{x})}(\mathbf{z})$ are random variables that are functions of the random variable $\mathbf{x}$ ($\mathbf{z}$ is marginalized over). Most existing work on the VAE models uses $q_\phi(\mathbf{z}|\mathbf{x})$ having the form $N(\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x}))$, where $\boldsymbol{\mu}_\phi(\mathbf{x})$ and $\boldsymbol{\Sigma}_\phi(\mathbf{x})$ are the outputs of a deep neural net parameterized by ϕ. In this case Eq. (4) reduces to $\mathrm{Cov}_{q_\phi(\mathbf{z})}(\mathbf{z}) = \mathbb{E}_{p(\mathbf{x})}\boldsymbol{\Sigma}_\phi(\mathbf{x}) + \mathrm{Cov}_{p(\mathbf{x})}(\boldsymbol{\mu}_\phi(\mathbf{x}))$, which we want to be close to an identity matrix. For simplicity, we choose entry-wise squared ℓ_2-norm as the measure of proximity. However, as the entanglement is mainly reflected in the off-diagonal entries of this matrix, we opt for two separate hyperparameters controlling the relative importance of the loss on the diagonal and off-diagonal entries. This gives rise to the following optimization problem for inference:

$$\max_{\theta,\phi} \mathrm{ELBO}(\theta, \phi) - \lambda_{od}\sum_{i\neq j}\left[\mathrm{Cov}_{p(\mathbf{x})}(\boldsymbol{\mu}_\phi(\mathbf{x}))\right]_{ij}^2 - \lambda_d\sum_i\left(\left[\mathrm{Cov}_{p(\mathbf{x})}(\boldsymbol{\mu}_\phi(\mathbf{x}))\right]_{ii} - 1\right)^2. \tag{5}$$

The regularization terms involving $\mathrm{Cov}_{p(\mathbf{x})}(\boldsymbol{\mu}_\phi(\mathbf{x}))$ in the above objective (5) can be efficiently optimized using SGD. We maintain a running estimate of $\mathrm{Cov}_{p(\mathbf{x})}(\boldsymbol{\mu}_\phi(\mathbf{x}))$ which is updated with every minibatch of $\mathbf{x} \sim p(\mathbf{x})$. The gradient for the current minibatch can be computed by treating the previous estimate of $\mathrm{Cov}_{p(\mathbf{x})}(\boldsymbol{\mu}_\phi(\mathbf{x}))$ as constant.

2.3 Comparison with β-VAE

Recently proposed β-VAE [7] proposes to modify the ELBO by upweighting the $\mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))$ term in order to encourage the inference of disentangled factors:

$$\max_{\theta,\phi} \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \beta\,\mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))\right], \tag{6}$$

where β is taken to be great than 1. Higher β is argued to encourage disentanglement at the cost of reconstruction error (the likelihood term in the ELBO). Authors report empirical results with β ranging from 4 to 250 depending on the dataset. As already mentioned, most VAE models proposed in the literature, including β-VAE, work with $N(\mathbf{0}, \mathbf{I})$ as the prior $p(\mathbf{z})$ and $N(\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x}))$ with diagonal $\boldsymbol{\Sigma}_\phi(\mathbf{x})$ as the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$. This reduces the objective (6) to

$$\max_{\theta,\phi} \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \frac{\beta}{2}\left(\sum_i\left(\left[\boldsymbol{\Sigma}_\phi(\mathbf{x})\right]_{ii} - \ln\left[\boldsymbol{\Sigma}_\phi(\mathbf{x})\right]_{ii}\right) + \|\boldsymbol{\mu}_\phi(\mathbf{x})\|_2^2\right)\right]. \tag{7}$$

Table 1: Disentanglement metric score [7] and reconstruction error (per pixel) on the test sets for 2D Shapes and CelebA ($\beta_1 = 4, \beta_2 = 60$ for 2D Shapes, and $\beta_1 = 4, \beta_2 = 8$ for CelebA)

Method	2D Shapes		CelebA	
	Metric	Reconst. error	Metric	Reconst. error
VAE	81.3	**0.0017**	7.5	**0.0876**
β-VAE (β=β_1)	80.7	0.0032	8.1	0.0937
β-VAE (β=β_2)	95.7	0.0113	7.1	0.1065
DIP-VAE	**98.7**	0.0018	**11.31**	0.0911

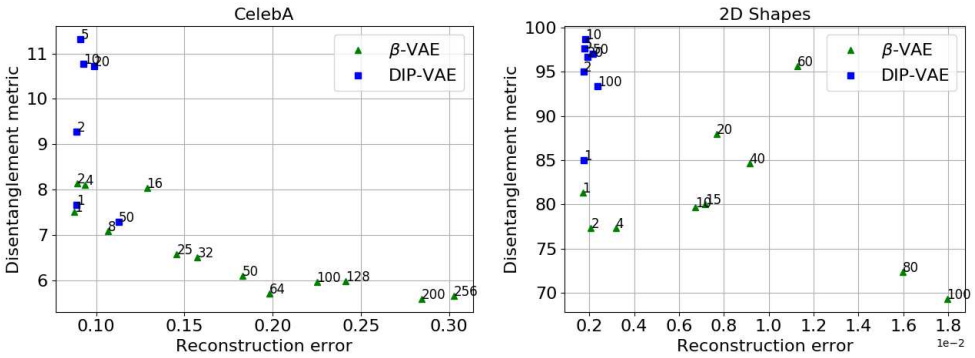


Figure 1: Disentanglement metric score [7] as a function of average reconstruction error (per pixel) on the test set for β-VAE and the proposed DIP-VAE (*left:* CelebA, *right:* 2D Shapes). For DIP-VAE, λ_d is set to $10\lambda_{od}$. The number next to each point in the plots is the value of β (β-VAE) or λ_{od} (proposed DIP-VAE).

For high values of β, β-VAE would try to pull $\boldsymbol{\mu}_\phi(\mathbf{x})$ towards zero and $\boldsymbol{\Sigma}_\phi(\mathbf{x})$ towards the identity matrix (as the minimum of $x - \ln x$ for $x > 0$ is at $x = 1$), thus making the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ insensitive to the observations. This is also reflected in the quality of the generated samples which is worse than VAE ($\beta = 1$), particularly for high values of β. Our proposed method does not have such increased tension between the likelihood term and the disentanglement objective, and the sample quality with our method is on par with the VAE.

Finally, we note that both β-VAE and our proposed method encourage disentanglement of inferred factors by pulling $\text{Cov}_{q_\phi(\mathbf{z})}(\mathbf{z})$ in Eq. (4) towards the identity matrix: β-VAE attempts to do it by making $\text{Cov}_{q_\phi(\mathbf{z}|\mathbf{x})}(\mathbf{z})$ close to $\mathbf{I}$ and $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}(\mathbf{z})$ close to $\mathbf{0}$ individually for all observations $\mathbf{x}$, while the proposed method directly works on $\text{Cov}_{q_\phi(\mathbf{z})}(\mathbf{z})$ (marginalizing over the observations $\mathbf{x}$) which retains the sensitivity of $q_\phi(\mathbf{z}|\mathbf{x})$ to the conditioned-upon observation.

3 Experiments

We evaluate our proposed method, referred as DIP-VAE subsequently (Disentangled Inferred Prior) on CelebA [13] which consists of celebrity face images, and 2D Shapes [7] which is a synthetic dataset of binary 2D shapes generated from the Cartesian product of the shape (heart, oval and square), x-position (32 values), y-position (32 values), scale (6 values) and rotation (40 values). We consider two baselines for the task of unsupervised inference of disentangled factors: (i) VAE [10, 17], and (ii) the recently proposed β-VAE [7]. To be consistent, we use the same CNN network architectures (for our encoder and decoder), and same latent dimensions as used in [7] for CelebA, 2D Shapes datasets. We fix $\lambda_d = 10\lambda_{od}$ in all our experiments.

Higgins et al. [7] proposed a metric to evaluate the disentanglement performance of the inference mechanism. Table 1 shows the scores on this disentanglement metric along with reconstruction error (which directly corresponds to the data likelihood) for the test sets for CelebA and 2D Shapes. It is evident that the proposed DIP-VAE outperforms β-VAE in both aspects[1]. Further we also show

[1]We get a lower metric score for β-VAE compared to what is reported in [7]. This could be due to a different evaluation protocol in [7] that trained 30 β-VAE models with different random seeds and *"discarded the bottom 50% of the thirty resulting scores and reported the remaining results"* (quoting verbatim from [7]).

the plot of how the disentanglement metric changes with the reconstruction error as we vary the hyperparameter for both methods (β and λ_{od}, respectively) in Fig. 1. It is clear that the proposed method gives much higher disentanglement metric score at little to no cost on the reconstruction error when compared with VAE ($\beta = 1$). The reconstruction error for β-VAE gets much worse as β is increased.

4 Future Work

An interesting direction for future work is to take into account the sampling biases in the generative process, both natural (e.g., sampling the *female* gender makes it unlikely to sample *beard* for face images in CelebA) as well as artificial (e.g., a collection of face images that contain much more smiling faces for males than females misleading us to believe $p(\text{gender,smile}) \neq p(\text{gender})p(\text{smile})$), which makes the problem challenging and also somewhat less well defined (at least in the case of natural biases).

References

[1] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[2] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 2017.

[3] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016.

[4] M. C. Fu. Gradient estimation. *Handbooks in operations research and management science*, 13:575–616, 2006.

[5] S. Gershman and N. Goodman. Amortized inference in probabilistic reasoning. In *Proceedings of the Cognitive Science Society*, volume 36, 2014.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, 2014.

[7] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

[8] G. E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer, 2011.

[9] T. Karaletsos, S. Belongie, and G. Rätsch. Bayesian representation learning with oracle constraints. *arXiv preprint arXiv:1506.05011*, 2015.

[10] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[11] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547, 2015.

[12] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, pages 1–101, 2016.

[13] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3730–3738, 2015.

[14] X. Nguyen, M. J. Wainwright, and M. I. Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, 2010.

[15] S. Nowozin, B. Cseke, and R. Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pages 271–279, 2016.

[16] S. Reed, K. Sohn, Y. Zhang, and H. Lee. Learning to disentangle factors of variation with manifold interaction. In *International Conference on Machine Learning*, pages 1431–1439, 2014.

[17] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[18] K. Ridgeway. A survey of inductive biases for factorial representation-learning. *arXiv preprint arXiv:1612.05299*, 2016.

[19] A. Stuhlmüller, J. Taylor, and N. Goodman. Learning stochastic inverses. In *Advances in neural information processing systems*, pages 3048–3056, 2013.

[20] T. Van Erven and P. Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

[21] J. Yang, S. E. Reed, M.-H. Yang, and H. Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In *Advances in Neural Information Processing Systems*, 2015.

Disentangling Dynamics and Content for Control and Planning

Ershad Banijamali[1] **, Ahmad Khajenezhad**[2]**, Ali Ghodsi**[3] **, Mohammad Ghavamzadeh**[4]
[1]School of Computer Science, University of Waterloo
[2]Sharif University of Technology
[3]Department of Statistics and Actuarial Science, University of Waterloo
[4]DeepMind
sbanijam@uwaterloo.ca, khajenezhad@ce.sharif.edu
aghodsib@uwaterloo.ca, ghavamza@google.com

Abstract

In this paper, We study the problem of learning a controllable representation for high-dimensional observations of dynamical systems. Specifically, we consider a situation where there are multiple sets of observations of dynamical systems with identical underlying dynamics. Only one of these sets has information about the effect of actions on the observation and the rest are just some random observations of the system. Our goal is to utilize the information in that one set and find a representation for the other sets that can be used for planning and ling-term prediction.

1 Introduction

The world surrounding us is full of events that we only observe them through high-dimensional sensory data. However, in many cases, these events can be described by few features and simple relations. Discovering the simple low-dimensional feature space is an underlying task in many data processing algorithms. With the recent advances in the area of artificial neural networks, use of deep structures for learning the low-dimensional representations has been outstandingly increased in different applications. A good representation is defined based on the task in hand.

In the area of control, a good representation means a low-dimensional feature space, in which the relation between different states of the system can be modeled by simple functions. Finding such representation has been studied recently in different works [2]. Deep autoencoders have been used for obtaining an appropriate representation for control in [5, 8]. This problem has been also studied in action respecting embedding (ARE) framework [3]. Embed to control (E2C) [9], finds a low-dimensional locally-linear embedding of the observations that allows planning and long-term prediction by applying model predictive controllers, e.g. iterative linear quadratic regulator (iLQR). More recently, robust controllable embedding (RCE), [1], has been proposed, which can handle noise in the dynamics of the system.

In this paper, we address this problem in a more generalized setting. Suppose we have different sets of high-dimensional observations from the systems that have the same underlying dynamics. Therefore, in all of the observations there exist a common set of features that correspond to the dynamics of the system. Our goal is to extract this set of features using only one set of observations and use the learned dynamics to do planning and long-term prediction for the other sets. To do so, we design a model that disentangles the features that contribute in dynamics and those who just contribute in the content of the image. Building such model requires dynamics information (i.e.

31st Conference on Neural Information Processing Systems (NIPS 2017), Workshop on Learning Disentangled Representations: from Perception to Control, Long Beach, CA, USA.

knowing how the actions change our observation from the system) in one set and there is no need to have such information in other sets.

Learning disentangled features has various applications in image and video processing and text analysis and has been studied in different works [6]. More recently, authors in [7, 4] proposed a model in the framework of generative adversarial networks (GANs) that disentangles dynamics and content for video generation. However, to the best of our knowledge, our model is the first model that proposes disentangling dynamics and content for control, planning, and prediction.

2 Problem Statement

Suppose we have different sets of high-dimensional observations from the states of dynamical systems where the underlying dynamics of the systems is the same. For now, let us assume that we only have one dynamical system and there are just two observation sets from this system from different angles. We make this assumption just for the sake of simplicity in notations, but it can be easily relaxed. The two observation sets are denoted by X and Y that belong to the observation spaces $\mathcal{X}$ and $\mathcal{Y}$, respectively.

Let us denote by $\mathcal{S}$, the true state space of the system, in which $\mathbf{s}_t$ represents the state of the system at time step t. The dynamics of the system in this space is defined by $f_{\mathcal{S}}$:

$$\mathbf{s}_{t+1} = f_{\mathcal{S}}(\mathbf{s}_t, \mathbf{u}_t) + \mathbf{n}^{\mathcal{S}} \tag{1}$$

where $\mathbf{n}^{\mathcal{S}}$ is the noise in the state space. We do not have any information about the state space and want to estimate it based on our observations.

Suppose set X consists of triples $(\mathbf{x}_t, \mathbf{u}_t, \mathbf{x}_{t+1})$, i.e. observation of the system at time t, action that is applied to the system at time t, and the next observation after applying $\mathbf{u}_t$ to the system, respectively. Therefore, we know how the actions change our observations in X. We also assume that the observations in this set have Markov property. Set Y also has some observations of the system from a different point of view. However, there is no information about the actions and the effect of the actions on our observation in this set. We denote the observations in this set by $\mathbf{y}_t$. Note that $\mathbf{x}_t$ and $\mathbf{y}_t$ are two different observations of the state $\mathbf{s}_t$. Since X and Y, are observations from one system, the underlying dynamics is the same. Suppose that our goal is to do planning and long-term prediction in $\mathcal{Y}$. Our approach to achieve this goal is to extract the dynamics information from X and leverage this information to build a model for Y.

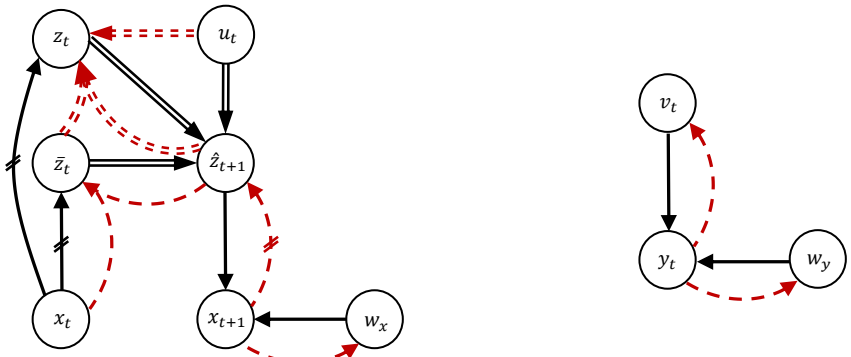
3 Model Description

There has been some efforts in finding a representation for high-dimensional observations of dynamical systems that is suitable for planning using neural networks. Recently, Robust Controllable Embedding (RCE) [1] has been proposed that shows good performance on this task. The RCE model is based on introducing a graphical model for the problem that describes the relation between pairs of observations and their embedded representations. Using deep variational learning, the lower bound of the conditional distribution of the observations is maximized.

We build our model up on RCE . However, instead of using only one latent variable, we assume that there are two independent variables in the latent space. One of these variables is related to the dynamics of the system and the other one is related to the content of the observation. Therefore we aim to disentangle the dynamics and content in the latent space. Such disentanglement allows us to model the dynamics of the observations, even though the content of them might be very different. Consider the graphical models in Fig. 1. Fig. 1a shows the model for X. In this figure, $\mathbf{z}_t$ and $\mathbf{w}_{\mathbf{x}}$ are the two latent variables that we want to represent the dynamics and content information, respectively. Similar to RCE, we want to have locally-linear dynamics in the latent space, i.e.:

$$\hat{\mathbf{z}}_{t+1} = \mathbf{A}_t \mathbf{z}_t + \mathbf{B}_t \mathbf{u}_t + \mathbf{c}_t \tag{2}$$

where $\mathbf{A}_t$, $\mathbf{B}_t$, and $\mathbf{c}_t$ are matrices that are learned during training the model. Building this locally-linear model will allow us to use iLQR method for control. We use $\mathbf{z}$ and $\hat{\mathbf{z}}$ to distinguish between encoding of $\mathbf{x}$ and the variable after transition. Fig. 1b shows the model for Y. This set is encoded with two latent variables $\mathbf{v}_t$ and $\mathbf{w}_{\mathbf{y}}$, representing dynamics and content, respectively. We would



(a) (b)

Figure 1: Graphical models. The black arrows are generative links and the red dashed ones are recognition links. The parallel lines show the deterministic links. **(a)** Graphical model for set X. $\bar{\mathbf{z}}_t$ and $\mathbf{z}_t$ are two samples from $p(\mathbf{z}_t|\mathbf{x}_t)$. The neural networks that parameterize the links with hatch marks are hard tied, i.e. $p(\mathbf{z}_t|\mathbf{x}_t) = p(\bar{\mathbf{z}}_t|\mathbf{x}_t) = q(\hat{\mathbf{z}}_t|\mathbf{x}_t)$. **(b)** Graphical model for Y

like to have a locally-linear dynamics similar to Eq. 2 for $\mathbf{v}$. All of the conditional distribution on these graphical models are parameterized by neural networks.

The goal in this work can be interpreted as maximizing the likelihood of observations, while imposing a further constraint that if $\mathbf{x}_t$ and $\mathbf{y}_t$ are two high-dimensional observations of the same state of the dynamical system(s), then we want $q(\mathbf{z}_t|\mathbf{x}_t)$ and $q(\mathbf{v}_t|\mathbf{y}_t)$ be close to each other, e.g. have small KL divergence.

Suppose $q^\star = q(\mathbf{z}_t, \bar{\mathbf{z}}_t, \hat{\mathbf{z}}_{t+1}, \mathbf{w}_\mathbf{x}|\mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{u}_t)$ and $q^\dagger = q(\mathbf{v}_t, \mathbf{w}_\mathbf{y}|\mathbf{y}_t)$. Based on the graphical model we can consider these factorizations for $q^\star$ and $q^\dagger$:

$$\begin{aligned} q^\star &= q_\phi(\mathbf{w}_\mathbf{x}|\mathbf{x}_{t+1}) q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1}) q_\varphi(\bar{\mathbf{z}}_t|\hat{\mathbf{z}}_{t+1}, \mathbf{x}_t) \delta(\mathbf{z}_t|\hat{\mathbf{z}}_{t+1}, \bar{\mathbf{z}}_t, \mathbf{u}_t) \\ q^\dagger &= q_\phi(\mathbf{w}_\mathbf{y}|\mathbf{y}_t) q_\phi(\mathbf{v}|\mathbf{y}_t) \end{aligned} \tag{3}$$

where ϕ and φ stand for encoder and transition network parameters, respectively. We also have the following factorization for the generative links in the graphical model:

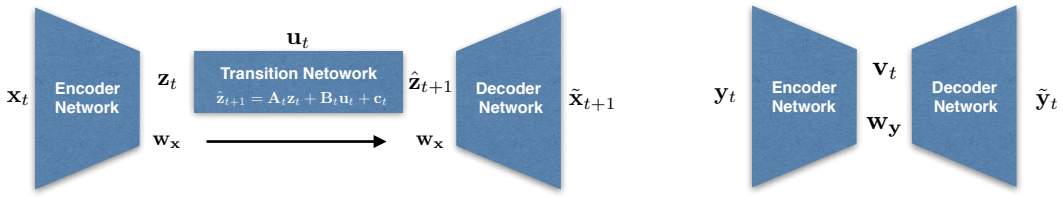
$$p(\mathbf{x}_{t+1}, \mathbf{z}_t, \bar{\mathbf{z}}_t, \hat{\mathbf{z}}_{t+1}, \mathbf{w}_\mathbf{x}|\mathbf{x}_t, \mathbf{u}_t) = p(\bar{\mathbf{z}}_t|\mathbf{x}_t) p(\mathbf{z}_t|\mathbf{x}_t) \delta(\hat{\mathbf{z}}_{t+1}|\bar{\mathbf{z}}_t, \mathbf{z}_t, \mathbf{u}_t) p(\mathbf{x}_{t+1}|\hat{\mathbf{z}}_{t+1}, \mathbf{w}_\mathbf{x}) p(\mathbf{w}_\mathbf{x}) \tag{4}$$

In this model, we want to maximize the likelihood of all the observations. Since we consider Markov property for set X, maximizing the likelihood of observations in X boils down to log-likelihood of the conditional distribution of the pair of observations. Therefore we will have:

$$\begin{aligned} &\log p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) + \log p(\mathbf{y}_t) \\ &\geq \mathbb{E}_{q^\star}\big[\log p(\mathbf{x}_{t+1}, \mathbf{z}_t, \bar{\mathbf{z}}_t, \hat{\mathbf{z}}_{t+1}, \mathbf{w}_\mathbf{x}|\mathbf{x}_t, \mathbf{u}_t) - \log q^\star\big] + \mathbb{E}_{q^\dagger}\big[\log p(\mathbf{y}_t, \mathbf{v}_t, \mathbf{w}_\mathbf{y}) - \log q^\dagger\big] \\ &= \mathbb{E}_{\substack{q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1}) \\ q_\phi(\mathbf{w}_\mathbf{x}|\mathbf{x}_{t+1})}}\big[\log p(\mathbf{x}_{t+1}|\hat{\mathbf{z}}_{t+1}, \mathbf{w}_\mathbf{x})\big] - \mathbb{E}_{q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1})}\big[\mathrm{KL}\big(q_\varphi(\bar{\mathbf{z}}_t|\hat{\mathbf{z}}_{t+1}, \mathbf{x}_t) \parallel p(\bar{\mathbf{z}}_t|\mathbf{x}_t)\big)\big] \\ &\quad + \mathrm{H}\big(q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1})\big) + \mathbb{E}_{\substack{q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1}) \\ q_\varphi(\bar{\mathbf{z}}_t|\mathbf{x}_t, \hat{\mathbf{z}}_{t+1})}}\big[\log p(\mathbf{z}_t|\mathbf{x}_t)\big] - \mathrm{KL}\big(q_\phi(\mathbf{w}_\mathbf{x}|\mathbf{x}_t) \parallel p(\mathbf{w}_\mathbf{x})\big) \\ &\quad + \mathbb{E}_{q^\dagger}\big[\log p(\mathbf{y}_\mathbf{t}|\mathbf{v}_t, \mathbf{w}_\mathbf{y})\big] - \mathrm{KL}\big(q_\phi(\mathbf{v}_t|\mathbf{y}_t) \parallel p(\mathbf{v}_t)\big) - \mathrm{KL}\big(q_\phi(\mathbf{w}_\mathbf{y}|\mathbf{y}_t) \parallel p(\mathbf{w}_\mathbf{y})\big) \end{aligned} \tag{5}$$

To maximize this lower bound we use the deep variational learning framework. We assume that the prior of the content variables, $\mathbf{w}_\mathbf{x}$ and $\mathbf{w}_\mathbf{y}$, are Gaussian. Also we assume $p(\bar{\mathbf{z}}_t|\mathbf{x}_t)$ is Gaussian. The constraint of minimizing the KL divergence between $q(\mathbf{z}_t|\mathbf{x}_t)$ and $q(\mathbf{v}_t|\mathbf{y}_t)$ can be imposed by considering $q(\mathbf{z}_t|\mathbf{x}_t)$ as the prior for $p(\mathbf{v}_t)$, i.e.:

$$p(\mathbf{v}_t) = \mathcal{N}\big(\mu_\phi(\mathbf{x}_t), \sigma_\phi(\mathbf{x}_t)\big) \tag{6}$$



(a) (b)

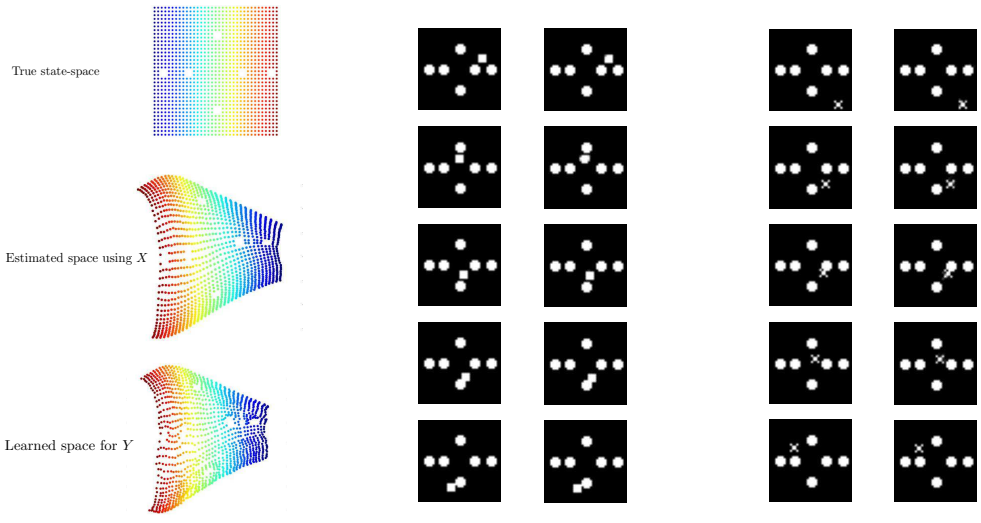
Figure 2: Networks of the model

Fig. 2 shows the high-level depiction of the networks in our model. In the case we use same networks for encoding and decoding the two observation sets (for example when the contents do not differ too much) , we can assume that $p(\mathbf{w_x})$ and $p(\mathbf{w_y})$ are two Gaussian distributions with different means.

4 Experiment Result

To evaluate the effectiveness of the proposed model, we consider the planar system domain. Consider an agent in a surrounded area, whose goal is to navigate from a corner to the opposite one, while avoiding the six obstacles in this area. The system is observed through a set of 40×40 pixel images taken from the top, which specify the agent's location in the area. Actions are two-dimensional and specify the direction of the agent's movement. Suppose that the difference between the two observation sets from this system is in the shape of the agent, as shown in Fig. 3. We use the same encoder and decoder for the two observation sets. We used 8000 samples (triples $(\mathbf{x}_t, \mathbf{u}_t, \mathbf{x}_{t+1})$) in the set X and only 2000 samples in set Y.

Fig. 3 shows the true map of the state-space of this system and the maps that are estimated using the model for the two observation sets. As we can see, the map that has been discovered using the information in X is very well preserved for the set Y. In this figure we can also see some predictions of the position of the agent for both sets given some actions versus the true position of the agent after applying those action. This shows that the model is successful in learning the dynamics for Y even though we did not have any information about the dynamics in this set.



(a) (b) (c)

Figure 3: **(a)** Top: The true state space of the system. Middle: estimated locally-linear latent space from set X. Bottom: The hidden space learned for set Y. **(b)**: Left: An initial observation from X on top and its next observations after applying four random actions Right: Reconstruction of the initial state and prediction of the next observations. **(c)**: Left: An initial observation from Y on top and its next observations after applying four random actions Right: Reconstruction of the initial state and prediction of the next observations

Table 1: Planar System

Dataset	Reconstruction Loss	Prediction Loss	Planning Loss	Success Rate
with action (X)	3.6 ± 1.7	6.2 ± 2.8	21.4 ± 2.9	100%
without action (Y)	3.9 ± 2.2	6.3 ± 3.0	22.0 ± 2.4	100%

To evaluate the performance of the model in planning, we provide different sets of initial and final observations in $\mathcal{X}$ and $\mathcal{Y}$, and use the learned models to find the policy that leads the agent to reach the final observation within T steps. We present the performance of the model in table 1 in terms of: **1)** *Reconstruction Loss* is the loss in reconstructing current observation using the encoder and decoder. **2)** *Prediction Loss* is the loss in predicting next observations, given current observation and current action, using the encoder, decoder, and transition network. **3)** *Planning Loss* is computed based on the following quadratic loss:

$$J = \sum_{t=1}^{T} (\mathbf{s}_t - \mathbf{s}^f)^\top \mathbf{Q} (\mathbf{s}_t - \mathbf{s}^f) + \mathbf{u}_t^\top \mathbf{R} \mathbf{u}_t. \tag{7}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are cost weighting matrices. $\mathbf{s}^f$ is the state corresponding to the final observation. We apply the sequence of actions returned by iLQR to the dynamical system and report the value of the loss in Eq. 7. **4)** *Success Rate* shows the number of times the agents reaches the goal within the planning horizon T, and remains near the goal in case it reaches it in less than T steps. For each of the sets, all the results are averaged over 20 runs.

5 Discussion

This model has potential applications in self-driving cars. Self-driving cars use many sensors to observe the surrounding environment that includes expensive sensors for dynamics estimation. They also use multiple cameras to monitor the area. Observations from the camera are rich in term of information about the content (objects in the area), however, extracting dynamics information using these observations is a hard task. On the other hand, the dynamics estimator sensors are poor in terms of the content information but provide information about action-state space with high accuracy. If we can find a way to transfer the learned dynamics from the sensor to the cameras, we can remove the sensor at the test time and reduce the cost of experiments.

References

[1] E. Banijamali, R. Shu, M. Ghavamzadeh, H. Bui, and A. Ghodsi. Robust locally-linear controllable embedding. In *arXiv preprint arXiv:1710.05373*, 2017.

[2] W. Böhmer, J. Springenberg, J. Boedecker, M. Riedmiller, and K. Obermayer. Autonomous learning of state representations for control: An emerging field aims to autonomously learn state representations for reinforcement learning agents from their real-world sensor observations. *Künstliche Intelligenz*, 29(4):353–362, 2015.

[3] M. Bowling, A. Ghodsi, and D. Wilkinson. Action respecting embedding. In *Proceedings of the 22nd international conference on Machine learning*, pages 65–72, 2005.

[4] E. Denton and V. Birodkar. Unsupervised learning of disentangled representations from video. *arXiv preprint arXiv:1705.10915*, 2017.

[5] S. Lange and M. Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–8, 2010.

[6] M. F. Mathieu, J. J. Zhao, J. Zhao, A. Ramesh, P. Sprechmann, and Y. LeCun. Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Information Processing Systems*, pages 5040–5048, 2016.

[7] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. *arXiv preprint arXiv:1707.04993*, 2017.

[8] N. Wahlström, T. Schön, and M. Desienroth. From pixels to torques: Policy learning with deep dynamical models. In *arXiv preprint arXiv:1502.02251*, 2015.

[9] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems*, pages 2746–2754, 2015.

Improved Neural Text Attribute Transfer with Non-parallel Data

Igor Melnyk* **Cicero Nogueira dos Santos** **Kahini Wadhawan** **Inkit Padhi**

Abhishek Kumar

IBM Research AI
T. J. Watson Research Center
Yorktown Heights, NY

Abstract

Text attribute transfer using non-parallel data requires methods that can perform disentanglement of content and linguistic attributes. In this work, we propose different improvements that enable the encode-decode framework to cope with text attribute transfer from non-parallel data. We perform experiments on the sentiment transfer task using two different datasets. For both datasets, our proposed method outperforms a strong baseline in two of the three employed evaluation metrics.

1 Introduction

The goal of the *text attribute transfer* task is to change an input text such that the value of a particular linguistic attribute of interest (e.g. language = English, sentiment = Positive) is transferred to a different desired value (e.g. language = French, sentiment = Negative). This task needs approaches that can disentangle the content from other linguistic attributes of the text. The success of neural encoder-decoder methods to perform text attribute transfer for the tasks of machine translation and text summarization rely on the use of large parallel datasets that are expensive to be produced. The effective use of non-parallel data to perform this family of problems is still an open problem.

In text attribute transfer from non-parallel data, given two large sets of non-parallel texts X_0 and X_1, which contain different attribute values s_0 and s_1, respectively, the task consists in using the data to train models that can rewrite a text from X_0 such that the resulting text has attribute value s_1, and vice-versa. The overall message contained in the rewritten text must be relatively the same of the original one, only the chosen attribute value should change. Two of the main challenges when using non-parallel data to perform such task are: (a) there is no straightforward way to train the encoder-decoder because we can not use maximum likelihood estimation on the transferred text due to lack of ground truth; (b) it is difficult to preserve content while transferring the input to the new style. Recent work from Shen et al. [9] showed promising results on style-transfer from non-parallel text by tackling challenging (a).

In this work, we propose a new method to perform text attribute transfer that tackles both challenges (a) and (b). We cope with (a) by using a single collaborative classifier, as an alternative to commonly used adversarial discriminators, e.g., as in [9]. Note that a potential extension to a problem of multiple attributes transfer would still use a single classifier, while in [9] this may require as many discriminators as the number of attributes. We approach (b) with a set of constraints, including the attention mechanism combined with cyclical loss and a novel noun preservation loss to ensure proper content transfer. We compared our algorithm with Shen et al. [9] on the sentiment transfer task on two datasets using three evaluation metrics (sentiment transfer accuracy, a novel *content preservation* metric and a perplexity), outperforming the baseline in terms of the first two.

*Corresponding author. Email: igor.melnyk@ibm.com

Workshop on Learning Disentangled Representations (NIPS 2017), Long Beach, CA, USA.

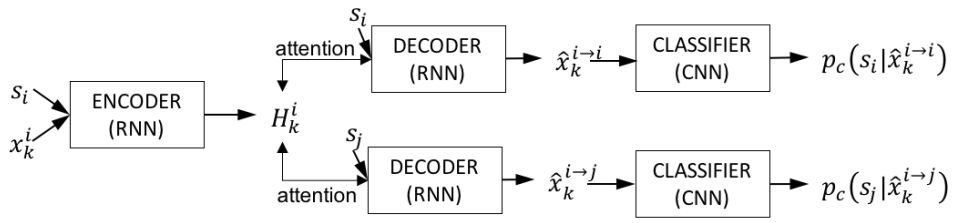


Figure 1: Proposed framework of a Neural Text Attribute Transfer algorithm using non-parallel data.

2 Proposed Method

We assume access to a text dataset consisting of two non-parallel corpora $X = X_0 \cup X_1$ with different attribute values s_0 and s_1 of a total of $N = m + n$ sentences, where $|X_0| = m$ and $|X_1| = n$. We denote a randomly sampled sentence k of attribute s_i from X as x_k^i, for $k \in 1, \ldots, N$ and $i \in \{0, 1\}$. A natural approach to perform text attribute transfer is to use a regular encoder-decoder network, however, the training of such network requires parallel data. Since in this work we consider a problem of attribute transfer on non-parallel data, we propose to extend the basic encoder-decoder by introducing a collaborative classifier and a set of specialized loss functions that enable the training on such data. Figure 1 shows an overview of the proposed attribute transfer approach.

The encoder (in the form of RNN), $E(x_k^i, s_i) = H_k^i$, takes as input a sentence x_k^i together with its attribute label s_i, and outputs H_k^i, a sequence of hidden states. The decoder/generator (also in the form of RNN), $G(H_k^i, s_j) = \hat{x}_k^{i\to j}$ for $i, j \in 0, 1$, takes as input the previously computed H_k^i and a desired attribute label s_j and outputs a sentence $\hat{x}_k^{i\to j}$, which is the original sentence but transferred from attribute value i to attribute value j. The hidden states H_k^i are used by the decoder in the attention mechanism [7, 2], and in general can improve the quality of the decoded sentence. For $i = j$, the decoded sentence $\hat{x}_k^{i\to i}$ is in its original attribute s_i (top part of Figure 1); for $i \neq j$, the decoded/transferred sentence $\hat{x}_k^{i\to j}$ is in a different attribute s_j (bottom part of Figure1). Denote all transferred sentences as $\hat{X} = \{\hat{x}_k^{i\to j} \mid i \neq j, k = 1, \ldots, N\}$. The classifier (in the form of CNN), then takes as input the decoded sentences and outputs a probability distribution over the attribute labels, i.e., $C(\hat{x}_k^{i\to j}) = p_C(s_j|\hat{x}_k^{i\to j})$ (see Eq. (3) for more details). By using the collaborative classifier our goal is to produce a training signal that indicates the effectiveness of the current decoder on transferring a sentence to a given attribute value.

Note that the top branch of Figure 1 can be considered as an auto-encoder and therefore we can enforce the closeness between $\hat{x}_k^{i\to i}$ and x_k^i by using a standard cross-entropy loss (see (1) below). However, for the bottom branch, due to lack of parallel data, we cannot use the same approach, and for this purpose we proposed a novel content preservation loss (see Eq. (2)). Finally, note that once we transferred X to $\hat{X}$ (forward-transfer step), we can now transfer $\hat{X}$ back to X (back-transfer step) by using the bottom branch in Figure 1 (see Eq. (5) and Eq. (6) below).

In what follows, we present the details of the loss functions employed in training of our model.

Algorithm 1 Training of the Neural Text Attribute Transfer Algorithm using Non-parallel Data.

Require: Two non-parallel corpora $X = X_0 \cup X_1$ with different attribute values s_0 and s_1.
Initialize $\theta_E, \theta_G, \theta_C$
repeat
— Sample a mini-batch of l original sentences $A = \{x_k^i\}_{k=1}^l$ from X, with $i \in \{0, 1\}$
— Sample a mini-batch of l transferred sentences $B = \{\hat{x}_k^{i\to j}\}_{k=1}^l$ from the generator's distribution p_G, where $\hat{x}_k^{i\to j} = G(E(x_k^i, s_i), s_j)$ with $i, j \in \{0, 1\}$
— Sample a mini-batch of l back-transferred sentences $C = \{\hat{x}_k^{i\to j\to i}\}_{k=1}^l$ from the generator's distribution p_G, where $\hat{x}_k^{i\to j\to i} = G(E(x_k^{i\to j}, s_j), s_i)$ with $i, j \in \{0, 1\}$
— Compute $\mathcal{L}_{\text{rec}}$ (1), $\mathcal{L}_{cnt\_rec}$ (2), $\mathcal{L}_{class\_td}$ (3), $\mathcal{L}_{class\_od}$ (4), $\mathcal{L}_{back\_rec}$ (5), and $\mathcal{L}_{class\_btd}$ (6)
— Update $\{\theta_E, \theta_G\theta_C\}$ by gradient descent on loss $\mathcal{L}(\theta_E, \theta_G, \theta_C)$ in Eq. (7)
until convergence

Reconstruction Loss. Given the encoded input sentence x_k^i and the decoded sentence $\hat{x}_k^{i\to i}$, the reconstruction loss measures how well the decoder G is able to reconstruct it:

$$\mathcal{L}_{rec} = \mathbb{E}_{(x_k^i, s_i)\sim X}\left[-\log p_G(\hat{x}_k^{i\to i}|E(x_k^i, s_i), s_i)\right]. \quad (1)$$

Content Preservation Loss. To enforce closeness between x_k^i and $\hat{x}_k^{i\to j}$ for $i \neq j$, we utilize the attention mechanism. Recall, that this mechanism enables to establish an approximate correspondence between the words in the original (encoded) and transferred (decoded) sentences. For example, denote the words in sentence x_k^i as $x_k^i = \{w_{kr}^i \mid r = 1, \ldots, |x_k^i|\}$ and similarly $\hat{x}_k^{i\to j} = \{w_{kr'}^{i\to j} \mid r' = 1, \ldots, |x_k^{i\to j}|\}$. Utilizing the attention mechanism, we can establish the correspondence (r, r') between the words. Among different pairings of such words we select only the ones where w_{kr}^i is a noun (e.g., as detected by a POS tagger),and enforce that the corresponding transferred word $w_{kr}^{i\to j}$ matches that noun, i.e.,

$$\mathcal{L}_{cnt\_rec} = \mathbb{E}_{(x_k^i=\{\ldots,w_{kr}^i,\ldots\}, s_i)\sim X}\left[-\log p_G\left(\hat{x}_k^{i\to j} = \{\ldots, w_{kr'}^{i\to j}\ldots\}|E(x_k^i, s_i), s_i\right)\right], \quad (2)$$

for indices r and r' such that w_{kr}^i is a noun and (r, r') is a pair established by attention mechanism. We note that although not always applicable, the above heuristic is very effective for attributes where the sentences can share the nouns (e.g., for sentiment transfer considered in Section 4).

Classification Loss - Transferred Data. The loss is formulated as follows:

$$\mathcal{L}_{class\_td} = \mathbb{E}_{(\hat{x}_k^{i\to j}, s_j)\sim \hat{X}}\left[-\ \log p_C(s_j|\hat{x}_k^{i\to j})\right]. \quad (3)$$

For the encoder-decoder this loss gives a feedback on the current generator's effectiveness on transferring sentences to a new attribute. For the classifier, it provides an additional training signal from generated data, enabling the classifier to be trained in a semi-supervised regime.

Classification Loss - Original Data. In order to enforce a high classification accuracy, the classifier also uses a supervised classification loss, measuring the classifier predictions on the original (supervised) instances $x_k^i \in X$:

$$\mathcal{L}_{class\_od} = \mathbb{E}_{(x_k^i, s_i)\sim X}\left[-\ \log p_C(s_i|x_k^i)\right]. \quad (4)$$

Back-transfer Reconstruction Loss. The *back-transfer (or cycle) loss* [10, 5] is motivated by the difficulty of imposing constraints on the transferred sentences. Back-transfer transforms the transferred sentences $\hat{x}_k^{i\to j}$ back to the original attribute s_i, i.e., $\hat{x}_k^{i\to j\to i}$ and compares them to x_k^i. This also implicitly imposes the constraints on the generated sentences and improves the content preservation (in addition to (2)). The loss is formulated as follows:

$$\mathcal{L}_{back\_rec} = \mathbb{E}_{(\hat{x}_k^{i\to j}, s_j)\sim \hat{X}}\left[-\ \log p_G(\hat{x}_k^{i\to j\to i}|E(\hat{x}_k^{i\to j}, s_j), s_i)\right], \quad (5)$$

which can be thought to be similar to an auto-encoder loss in (1) but in the attribute domain.

Classification Loss - Back-transferred Data: Finally, we ensure that the back-transferred sentences $\hat{x}_k^{i\to j\to i}$ have the correct attribute label s_i:

$$\mathcal{L}_{class\_btd} = \mathbb{E}_{(\hat{x}_k^{i\to j}, s_j)\sim \hat{X}}\left[-\ \log p_C(s_i|G(E(\hat{x}_k^{i\to j}, s_j), s_i))\right] \quad (6)$$

In summary, the training of the components of our architecture consists in optimizing the following loss function using stochastic gradient descent with back-propagation for some weights $\lambda_i > 0$:

$$\mathcal{L}(\theta_E, \theta_G, \theta_C) =$$
$$= \min_{E,G,C} \lambda_1 L_{rec} + \lambda_2 L_{cnt\_rec} + \lambda_3 \mathcal{L}_{back\_rec} + \lambda_4 \mathcal{L}_{class\_od} + \lambda_5 \mathcal{L}_{class\_td} + \lambda_6 \mathcal{L}_{class\_btd}. \quad (7)$$

The Algorithm 1 summarizes the above discussion and shows the main steps of the training of the proposed approach.

3 Related Work

Attribute transfer has been studied more extensively in the context of images than in the text domain, with several works studying the style transfer task under the setting of non-parallel data [4, 11].

However, style/attribute transfer in text is fundamentally different as textual data is sequential and of potentially varying lengths, versus constant-sized images. In the image domain, one of the similar works is CycleGAN [11], which also employs a cycle consistency loss (similar to our *back-transfer loss*) that ensures that composition of a transfer and its reverse is close to the identity map. However, there are several key differences between CycleGAN and our work: (i) we use of a single generator for generating both styles which makes it easier to scale to multiple style transfer, (ii) we use a collaborative classifier for measuring the style instead of a adversarial discriminator, which imparts stability to the training, (iii) additional syntactic regularizers for better content preservation.

Controlled text generation and style transfer without parallel data has also received attention from the language community recently [8, 6, 3, 9]. Ficler and Goldberg [3] consider the problem of attribute conditioned generation of text in a conditioned language modeling setting using LSTM. Mueller et al. [8] allows modifying the hidden representations to generate sentences with desired attributes which is measured by a classifier, however their model does not explicitly encourage content preservation. Our proposed model has some similarities with the approach taken by Hu et al. [6] and Shen et al. [9], with the main differences being that instead of VAE and adversarial discriminators we use a simple encode-decoder framework with a collaborative classifier augmented with the attention mechanism and a set of specially designed content preservation losses.

4 Experiments and Results

In this Section we present experimental results of applying the proposed approach for sentiment transfer as one example of text attribute transfer. We compared the algorithm with the approach of [9] on two datasets. One is the dataset from [9], which is based on Yelp restaurant reviews and contains (179K, 25K, 51K) sentences for (training, validation, testing) based on negative reviews and similarly (268K, 38K, 76K) positive sentences. The sentences had a maximum length of 17 words. The second dataset is based on general customer reviews on Amazon [1], from which we selected (265K, 33K, 33K) positive and the same number of negative sentences, each having up to 7 tokens per sentence.

Table 1: Evaluation results on Yelp and Amazon datasets. For Yelp, the pre-trained classifier had a default accuracy of 97.4% and the pre-trained language model had a default perplexity of 23.5. For Amazon, these values were 82.02% for classification and 25.5 for perplexity.

	Yelp			Amazon		
	Sentiment	Content	Perplexity	Sentiment	Content	Perplexity
Shen et. al [9]	86.5	38.3	27.0	32. 8	71.6	27.3
Our Method	94.4	77.1	80.1	59.5	77.5	43.7

Table 2: Examples of sentences transferred from positive to negative sentiment on Yelp dataset

Original	their food was definitely delicious	love the southwestern burger
Shen et. al [9]	there was so not spectacular	avoid the pizza sucks
Our Method	their food was never disgusting	avoid the grease burger
Original	restaurant is romantic and quiet	the facilities are amazing
Shen et. al [9]	the pizza is like we were disappointed	the drinks are gone
Our Method	restaurant is shame and unprofessional	the facilities are ridiculous

Table 3: Examples of sentences transferred from negative to positive sentiment on Yelp dataset

Original	sorry they closed so many stores	these people will try to screw you over
[9]	thanks and also are wonderful	these guys will go to work
Ours	amazing they had so many stores	these people will try to thank you special
Original	i wish i could give them zero stars	seriously , that 's just rude
[9]	i wish i love this place	clean , and delicious ...
Ours	i wish i 'll give them recommended stars	seriously , that 's always friendly

We used three evaluation metrics: (i) sentiment accuracy, which is computed based on pre-trained classifier (estimated on the training part of each dataset) and measures the percentage of sentences in the test set with correct sentiment label; (ii) content preservation accuracy, a new evaluation metric proposed in this work, which is computed as the percentage of the transferred sentences where each

of them has at least one of the nouns present in the original sentence; (iii) perplexity score, which is computed based on pre-trained language model (estimated on the training part of each dataset) and measures the quality of the generated text.

The results are presented in Table 1. As compared to the algorithm of Shen et. al [9], the proposed method although not able to get better perplexity scores, it can achieve more accurate sentiment transfer and better content preservation. A possible explanation of having higher perplexity is that since the algorithm of [9] does not explicitly enforce content similarity, it has an easier job of achieving high sentiment accuracy and low perplexity of the transferred sentences. Our algorithm, on the other hand, is penalized if the content changes, which forces it to sacrifice the perplexity. Achieving better results across all the metrics still remains a challenge.

In Table 3 we also show some of the sentences generated by both algorithms on Yelp dataset. The algorithm of [9], although able to create well structured sentences with correct sentiment labels, in many cases it cannot accurately preserve the content. On the other hand, our approach may generate text with somewhat higher perplexity but ensures a better sentiment and content transfer.

5 Conclusion

In this work we proposed a novel algorithm for text attribute transfer with non-parallel corpora based on the encoder-decoder architecture with attention, augmented with the collaborative classifier and a set of content preservation losses. Although the experimental evaluations showed promising results, a number of challenges remain: (i) achieve better results across all the three metrics and propose new evaluation metrics to better capture the quality of transfer; (ii) improve the architecture to enable transfer for more challenging text attributes (e.g., such as professional-colloquial) where the text goes under more significant transformation then in a simpler sentiment transfer tasks; (iii) extend the architecture to work in a multi-attribute transfer, a more challenging problem.

References

[1] Amazon reviews dataset. `https://www.kaggle.com/bittlingmayer/amazonreviews`. Accessed: 2017-11-05.

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] J. Ficler and Y. Goldberg. Controlling linguistic style aspects in neural language generation. *arXiv preprint arXiv:1707.02633*, 2017.

[4] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *CVPR*, pages 2414–2423, 2016.

[5] D. He, Y. Xia, T. Qin, L. Wang, N. Yu, T. Liu, and W. Ma. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, pages 820–828, 2016.

[6] Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing. Towards controllable generation of text. In *International Conference on Machine Learning*, 2017.

[7] T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, September 2015.

[8] J. Mueller, D. Gifford, and T. Jaakkola. Sequence to better sequence: continuous revision of combinatorial structures. In *ICML*, pages 2536–2544, 2017.

[9] T. Shen, T. Lei, R. Barzilay, and T. Jaakkola. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*, 2017.

[10] J. Zhu, T. Park, P. Isola, and A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.

[11] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.

JADE: Joint Autoencoders for Dis-Entanglement

Amir-Hossein Karimi[1]*, **Ershad Banijamali**[1]*, **Alexander Wong**[2], **Ali Ghodsi**[3]
[1]School of Computer Science, University of Waterloo
[2]Systems Design Engineering, University of Waterloo
[3]Department of Statistics and Actuarial Science, University of Waterloo
{a6karimi, sbanijam, a28wong, aghodsib}@uwaterloo.ca

Abstract

The problem of feature disentanglement has been explored in the literature, for the purpose of image and video processing and text analysis. State-of-the-art methods for disentangling feature representations rely on the presence of many labeled samples. In this work, we present a novel method for disentangling factors of variation in data-scarce regimes. Specifically, we explore the application of feature disentangling for the problem of supervised classification in a setting where few labeled samples exist, and there are no unlabeled samples for use in unsupervised training. Instead, a similar datasets exists which shares at least one direction of variation with the sample-constrained datasets. We train our model end-to-end using the framework of variational autoencoders and are able to experimentally demonstrate that using an auxiliary dataset with similar variation factors contribute positively to classification performance, yielding competitive results with the state-of-the-art in unsupervised learning.

1 Introduction

In machine learning, samples in a dataset originate via complicated processes driven by a number of underlying factors. Individual factors lead to independent directions of variations in the observed samples, while the accumulation of factors give rise to the rich structure characteristic of these datasets. The underlying factors often interact in complicated and unpredictable ways, and appear tightly *entangled* in the raw data. Being able to tease apart the effect of underlying factors is a fundamental challenge in understanding these datasets.

For instance, a dataset containing images of natural scenery may be subject to variation in lighting conditions, camera elevation, and the appearance of the scene itself. Controlling and restraining variation at data acquisition time is difficult, and limits the number of acceptable samples in the dataset. On the other hand, capturing annotations for every direction of variation is time-consuming and infeasible. Therefore, designing methods that automatically learn to separate out underlying factors (known and unknown) is relevant for many applications in machine learning.

One area that has enjoyed tremendous success for separating factors of variation is supervised learning. The representations learned here aim to satisfy a specific task that is driven by the explicit labels in the dataset. Therefore, these representations are invariant to factors of variation that are uninformative for solving the task at hand. For example, when identifying individuals in a school yearbook, the identity of the person is paramount compared to their facial expression. Hence, a simple method that simply discards the irrelevant variation in expression will perform really well. Learning invariant representations, however, require many samples and comes at the cost of needing to train a new model for a closely related task that depends on an alternative direction of variation.

*equal contribution

31st Conference on Neural Information Processing Systems (NIPS 2017), Workshop on Learning Disentangled Representations: from Perception to Control, Long Beach, CA, USA.

It would seem reasonable then to desire a strategy that captures all directions of variation in a single model in a *disentangled* manner allowing one to infer all factors for a given sample in the absence of labels for each factor.

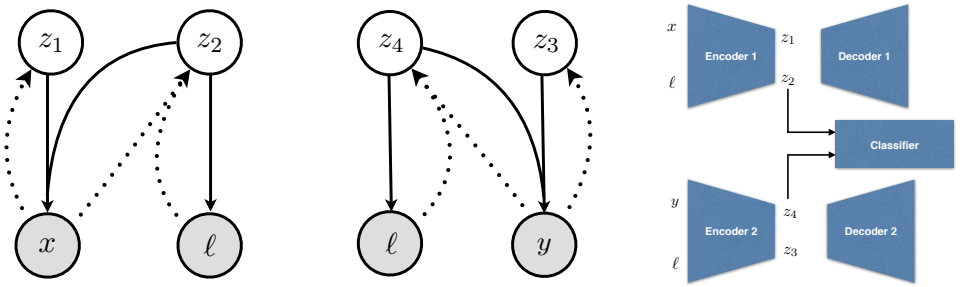
Current state-of-the-art strategies for disentangling factors of variation mostly fall victim to the challenges in deep learning and rely on the presence of abundant data samples. In [5], the authors were able to accurately separate out lighting, pose, and shape while sampling seemingly unlimitedly from an auxiliary generative model that creates samples with different variations. The results presented in [9, 7] also build upon datasets containing often hundreds of thousands of samples. Whereas [3, 11] use very few samples in their training process, these methods are semi-supervised and have access to unlabeled samples from the same dataset following the same statistical distribution.

In this work, we explore classification in a data-scarce scenario where not only are there few labeled samples available, there are also no unlabeled samples from which one could perform semi-supervised training. These situations commonly arise in medical imaging datasets, e.g., pancreatic cancer MRI images are scarce whereas breast cancer MRI images are abundant ([2] and references therein). In such a situation, we ask whether one can employ a secondary dataset, with many samples, similar content, but different style, to improve the performance of a benchmark classification model. What remains to be demonstrated is how to learn good intermediate representations that can be shared across tasks and use the disentanglement process of the secondary dataset to effectively disentangle the factors of variation in the primary dataset of interest. Essentially we are entangling together the feature disentangling of two similar datasets. This is the focus of the work below.

2 Model Description

In this work, we consider a situation where we are given a labeled dataset, X, with limited number of points. We denote the label variable by ℓ. We also have access to another dataset Y with a larger number of points that share the same categories as Y. However, the underlying distribution of the datasets are different. Let us denote the distribution for X and Y by $p(x)$ and $p(y)$, respectively. Suppose that our goal is to classify unseen data points that come from $p(x)$, i.e. to maximize $p(\ell|x)$. Building a classifier that simply uses X can lead to low accuracy and overfitting, due to its small size. Therefore we want to leverage the information of Y about the label variable and build a model that can classify the points from $p(x)$ with higher accuracy.

Our approach to address this problem is to disentangle the features in X and Y that contribute in predicting the label variable (i.e., content) from the features that contribute to the style of X and Y. Consider the graphical model in Fig. 1a. We assume there are two pairs of latent variables that describe each of x and y. Based on this figure, suppose that z_1 and z_2 generate samples in dataset X and z_3 and z_4 generated samples in dataset Y. If we assume that z_2 and z_4 are the latent variables that carry all the information about the label variable ℓ then $p(\ell|z_2) = p(\ell|z_4)$. Considering the same prior distributions over z_2 and z_4, i.e. $\mathcal{N}(0, I)$, we can guarantee the disentanglement of latent features by asserting that $p(z_2|\ell) = p(z_4|\ell)$. However, these posteriors are intractable. To approximate them we use the framework of variational inference where $p(z_2|\ell)$ and $p(z_4|\ell)$ are approximated by $q(z_2|x, \ell)$ and $q(z_4|y, \ell)$, respectively. Therefore, by matching these approximating distribution, we guarantee that only z_2 and z_4 carry information regarding the label ℓ (i.e., content) and therefore are disentangled from z_1 and z_3 respectively which represent style.



(a) (b)

Figure 1: (a) Graphical models of the method. (b) Network structure of the method

All the conditional distributions on the graphical models in Fig. 1a are parameterized by the neural networks depicted in Fig. 1b. The joint model here builds on earlier work in [10] where an autoencoder and a discriminator were trained in the framework of contractive discriminative analysis for semi-supervised learning. Here, we use the variational autoencoding [4] approach to jointly train two networks that simultaneously extract shared discriminative features present in the primary and secondary datasets. This architecture is reminiscent of Domain Separation Networks [1]. The proposed JADE model, however, focuses on a shared classifier for improved classification and joint disentanglement instead of a shared encoder and decoder.

The variational lower bound on the joint distribution of the observations is:

$$\begin{aligned}
\log p(x,\ell) \geq \mathcal{L}(x,\ell) &= \mathbb{E}_{\substack{q(z_1|x) \\ q(z_2|x,\ell)}} \big[\log p(x|z_1,z_2)\big] + \mathbb{E}_{q(z_2|x,\ell)}\big[\log p(\ell|z_2)\big] \\
&\quad -\mathrm{KL}\big(q(z_1|x) \parallel p(z_1)\big) - \mathrm{KL}\big(q(z_2|x,\ell) \parallel p(z_2)\big)\big) \\
\log p(y,\ell) \geq \mathcal{L}(y,\ell) &= \mathbb{E}_{\substack{q(z_3|y) \\ q(z_4|y\ell)}} \big[\log p(x|z_3,z_4)\big] + \mathbb{E}_{q(z_4|y,\ell)}\big[\log p(\ell|z_4)\big] \\
&\quad -\mathrm{KL}\big(q(z_3|x) \parallel p(z_3)\big) - \mathrm{KL}\big(q(z_4|x,\ell) \parallel p(z_4)\big)\big)
\end{aligned} \tag{1}$$

We would like to maximize the sum over the above lower bounds. The approximating distributions are from exponential family (Gaussian) and to match them we assume that for the samples that are from the same class in the two datasets, we want to minimize $\mathrm{KL}\big(q(z_2|x,\ell) \parallel q(z_4|y,\ell)\big)$. Given this condition, the overall objective of the model is:

$$\max_{\Theta} \mathcal{L}(x,\ell) + \mathcal{L}(y,\ell) - \mathrm{KL}\big(q(z_2|x,\ell) \parallel q(z_4|y,\ell)\big) \tag{2}$$

where Θ represents the entire parameter set of neural networks.

3 Experiments

Datasets: Our framework addresses the problem of performing supervised classification in data-scarce regimes where there exists a secondary dataset that has at least one direction of variation in common with the primary sample-constrained dataset. In our experiments we emulate this scenario with commonly used datasets such as MNIST [6] and SVHN [8]. Because MNIST is relatively easier to learn, even with very few samples, we select SVHN as the sample-constrained primary dataset that is difficult to learn, and use the entirety of MNIST as the secondary dataset. These datasets differ in appearance and style: whereas MNIST is gray-scale and comes in 28×28 pixel images, SVHN has three color channels and comes in 32×32 pixel images. However, both datasets represent the same content (i.e., digit values) across different styles. This similarity in content of both datasets is what makes MNIST a good secondary dataset to boost SVHN's supervised classification performance.

Model Comparison: To evaluate the performance of our framework, we first develop a benchmark for supervised classification of SVHN. Here, we choose a relatively powerful convolutional neural network (CNN) architecture combined with a multi-layer perceptron (MLP) as the supervised classification model. The CNN architecture comprises of 4 layers of 3×3 spatial convolutions ($\{64, 96, 64, 8\}$ filters respectively) followed by ReLU and interspersed with 3 layers of $2\times$ max-pooling. The MLP contains 3 blocks of 500-dimensional fully connected layers, followed by ReLU and Dropout ($p = 0.5$) layers [12]. A 10-dimensional bottleneck layer was placed in between the CNN and the MLP to encourage only important features from being retained. A final softmax layer is present at the end of the network for 10-way classification. The loss for this model is measured using categorical cross-entropy. This architecture is referred to as *single classifier* (i.e., benchmark).

A simple extension of above setup is a model that jointly trains SVHN and MNIST on a shared MLP classifiers using features extracted from separate CNN feature extractors, one per dataset. The CNN used for SVHN and the MLP follow the same architecture as the benchmark above. The CNN architecture for MNIST comprises of 3 layers of 3×3 spatial convolutions ($\{32, 32, 16\}$ filters respectively) followed by ReLU and interspersed with 3 layers of $2\times$ max-pooling. A 10-dimensional bottleneck layer was placed in between the CNN for MNIST and the shared MLP to capture the latent features of MNIST. Feature-extracted samples from both datasets are fed into the shared MLP in alternation and trained jointly. The loss of the system is the sum of the categorical cross-entropy losses for both datasets on the shared classifier. This setup is called *paired classifier*.

Table 1: Classification error rates for SVHN on limited data: 100 samples per each class. Error rates calculated using the entirety of SVHN's test set. Results of our experiments are averaged over 3 runs. We observe improved SVHN classification performance without sacrificing near state-of-the-art performance on MNIST.

Method	SVHN (1000 samples)	MNIST (45K samples)
VAE (M1+M2) [3]	36.02±0.10	-
Siddharth et al. [11]	28.71±2.38	-
Single Classifier (benchmark)	32.31±1.56	-
Paired Classifier	30.17±2.77	0.82±0.05
JADE (proposed)	29.08±0.92	0.72±0.03

Finally, the proposed model (outlined in Fig. 1b) extends upon the previous two methods by adding a decoder network to reconstruct the 10-dimensional latent representations from each of the CNN feature extractors. To encourage disentanglement of features in the latent space, and to perform factor separation in a way that the MLP classifier is only given content-related features (i.e., digit values), we increase the size of the latent spaces from 10 to 20 dimensions. However, only 10 of the latent dimensions resulting from each CNN are passed into the shared MLP, essentially keeping consistent with the previous method in terms of classifier capacity. All 20 latent dimensions are used to reconstruct the inputs via a decoder that identically mirrors the corresponding CNN ($2\times$ up-sampling layers used in place of $2\times$ max-pooling). Losses are defined in Section 2. Due to the autoencoding structure of this model, we refer to it as *JADE: Joint Autoencoders for Dis-Entanglement*.

Discussion: The results of our experiments have been presented in Table 1. Here we compare the results of the single classifier (i.e., benchmark model), paired classifier, and proposed model (JADE) alongside those from Kingma et al. [3] and Siddharth et al. [11]. It is worth pointing out that the former 3 models are trained only on 1000 labeled sample from SVHN, whereas the cited models use the remainder of the SVHN training dataset in an unsupervised fashion. We, on the other hand, use all of the MNIST dataset to train the paired classifier in JADE.

These results demonstrate that when dealing with sample-constrained regimes without unlabeled samples, one can use a similar dataset with at least one shared direction of variation to improve classification performance. This can be seen when comparing the performance of a single classifier (32.31±01.56) with that of a paired classifier (30.17±02.77). On top of this, we see that the JADE model which learns to jointly disentangle SVHN and MNIST features performs even better than the former methods, sitting at 29.08±00.92. This is in line with our hypothesis that only the directions of variation shared between MNIST and SVHN (i.e., content) will contribute positively to classification performance on SVHN, and other factors of variation should be disentangled.

We hypothesize that actively attempting to disentangle variation factors (i.e., in JADE) is better than allowing the network to attempt to discard uninformative factors (i.e., paired classifier) given the sample-constrained regime. To assert that the JADE setup is indeed disentangling variation factors, we conduct the following simple experiment: observe the variation in latent space values as different types of samples are passed into the network. In Fig. 2a, we have shown how latent activations change when the SVHN CNN is fed with 500 samples from the same class (i.e., same content but varying style). These activations are shown for the 20 latent parameters (of which only 10 are passed into the MLP classifier, and all used for reconstruction) across 10 classes of digits in MNIST. We observe that in this setup where content is fixed, the normalized variance of the latent variables that

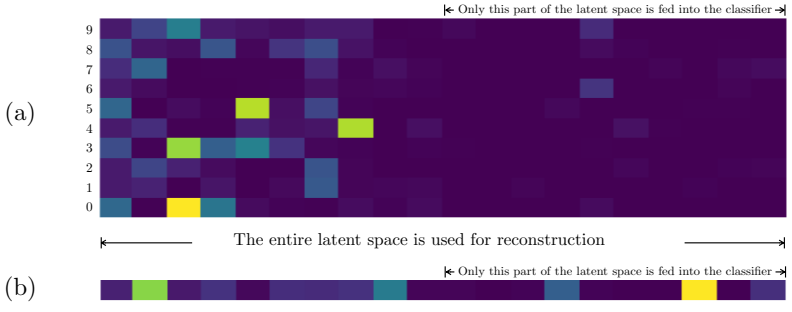


Figure 2: **(a)** variance normalized activations of latent space parameters, averaged over 500 random samples from each of 10 classes in SVHN; when content is fixed, the part of the latent space that feeds into the classifier exhibits weaker variance in activations compared to the part of the latent space that seemingly represents style over the 500 samples. **(b)** variance normalized activations of latent space parameters for 2500 random samples from SVHN spanning various style and content; all 20 latent space parameters fire for random splits of the data.

are fed into the MLP classifier is much lower than the variance of latent variables that are solely used for reconstruction. In Fig. 2b, we observe an interesting and complementary phenomena when we pass in 2500 randomly selected test samples into the SVHN CNN. Here, both the style and the content vary between input samples, and we observe that all 20 latent parameters are active given the varying input. These observations suggest that JADE is able to successfully disentanglement content and style in low-data SVHN using the help of MNIST as an auxiliary similar dataset.

4 Conclusion and Future Work

In this work, we explore the application of feature disentangling for the problem of supervised classification in a setting where few labeled samples exist, and there are no unlabeled samples for use in unsupervised training. Instead, a similar datasets exists which shares at least one direction of variation with the sample-constrained datasets. We train our model end-to-end using the framework of variational autoencoders and experimentally demonstrated that using a secondary dataset with similar content to SVHN leads to improvements in supervised classification performance.

Given the autoencoding structure of the proposed framework, a reasonable next step is to explore using an ensemble of auxiliary datasets, say one for content and another for style, to augment not only the classification power of the system, but also its reconstruction and generation ability. Currently, reconstruction quality is lacking as samples are being generated using the limited samples. Finally, an exciting extension of the JADE framework is cross-task or cross-modality data synthesis, e.g., learning a joint representation that captures high-level concepts for all modalities of the same object allows for bi-directional generation of missing modalities from the remaining modalities [13].

References

[1] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In *Advances in Neural Information Processing Systems*, pages 343–351, 2016.

[2] H. Greenspan, B. van Ginneken, and R. M. Summers. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, 35(5):1153–1159, 2016.

[3] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[4] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[5] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547, 2015.

[6] Y. LeCun, C. Cortes, and C. J. Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

[7] M. F. Mathieu, J. J. Zhao, J. Zhao, A. Ramesh, P. Sprechmann, and Y. LeCun. Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Information Processing Systems*, pages 5040–5048, 2016.

[8] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011.

[9] S. Reed, K. Sohn, Y. Zhang, and H. Lee. Learning to disentangle factors of variation with manifold interaction. In *International Conference on Machine Learning*, pages 1431–1439, 2014.

[10] S. Rifai, Y. Bengio, A. Courville, P. Vincent, and M. Mirza. Disentangling factors of variation for facial expression recognition. *Computer Vision–ECCV 2012*, pages 808–822, 2012.

[11] N. Siddharth, B. Paige, V. de Meent, A. Desmaison, F. Wood, N. D. Goodman, P. Kohli, P. H. Torr, et al. Learning disentangled representations with semi-supervised deep generative models. *arXiv preprint arXiv:1706.00400*, 2017.

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

[13] M. Suzuki, K. Nakayama, and Y. Matsuo. Joint multimodal learning with deep generative models. *arXiv preprint arXiv:1611.01891*, 2016.

Quantifying the Effects of Enforcing Disentanglement on Variational Autoencoders

Momchil Peychev, Petar Veličković, Pietro Liò
Department of Computer Science and Technology
University of Cambridge
mpeychev@cantab.net, {petar.velickovic, pietro.lio}@cst.cam.ac.uk

Abstract

The notion of disentangled autoencoders was proposed as an extension to the variational autoencoder by introducing a disentanglement parameter β, controlling the learning pressure put on the possible underlying latent representations. For certain values of β this kind of autoencoders is capable of encoding independent input generative factors in separate elements of the code, leading to a more interpretable and predictable model behaviour. In this paper we quantify the effects of the parameter β on the model performance and disentanglement. After training multiple models with the same value of β, we establish the existence of consistent variance in one of the disentanglement measures, proposed in literature. The negative consequences of the disentanglement to the autoencoder's discriminative ability are also asserted while varying the amount of examples available during training.

1 Introduction

The exponential growth in data availability and the rapid increase of computational power in the past decade have allowed neural network based algorithms to achieve impressive practical results in the fields of computer vision [14, 18], natural language processing and generation [7, 21], and game playing [19] to mention a few, surpassing human performance on several complex tasks [8, 19]. Despite the undeniable potential of the deep learning approach, however, more research is required to better understand its limits [20].

This work primarily concerns the model of a disentangled autoencoder which represents a recent development towards building more transparent and interpretable generative models. It is capable of learning independent generating factors separately in the network, thus being more predictable in its behaviour. Given certain input data we might know what values to expect for the code and, conversely, small disturbances of the code result in expected changes of the output. We study the properties of this model with respect to changing the values of the disentanglement parameter β, measuring both its disentanglement level and discriminative ability.

Autoencoders have been part of the neural network field since the late 80s [2, 12]. Because of their capability to perform dimensionality reduction, they are sometimes considered to do a more powerful non-linear Principal Component Analysis (PCA) [1, 6]. The disentangled autoencoder was initially introduced by Higgins *et al.* [9, 10] and has ever since been applied in semi-supervised learning environments as well [16]. It can be considered a generalisation of the variational autoencoder devised by Kingma and Welling [13]. Earlier attempts at disentangled factor learning were reported to either require a priori knowledge about the data generating factors [3, 11, 17], or do not scale well [4, 5].

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

2 Background

Kingma and Welling [13] derived the variational autoencoder framework by rearranging the evidence lower bound (ELBO) so that the intractable posterior $p_\theta(\mathbf{z}|\mathbf{x})$ can be eliminated

$$\mathrm{ELBO}(\theta, \phi) = \log p_\theta(\mathbf{x}) - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}|\mathbf{x})) \tag{1}$$

$$\mathrm{ELBO}(\theta, \phi) = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})) \tag{2}$$

The first term corresponds to the autoencoder's reconstruction error and the second one is the Kullback-Leibler (KL) divergence between the posterior approximation $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{z})$, acting as a regulariser. In practice, this cost is typically dominated by the reconstruction error so Higgins *et al.* [9, 10] took this approach further by specifying the optimisation problem

$$(\phi, \theta) = \max_{\phi,\theta} \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] \quad \text{subject to} \quad D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p_\theta(\mathbf{z})) < \epsilon \tag{3}$$

for $\epsilon > 0$. Applying the Karush-Kuhn-Tucker conditions [1], Equation (3) can be written as a Lagrangian

$$\mathcal{L}(\theta, \phi; \mathbf{x}) = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta\, D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p_\theta(\mathbf{z})) \tag{4}$$

with $\beta \geq 0$, deriving the final disentangled autoencoder cost function. A practical choice is to set $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$ and $p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$. In this way not only the D_{KL} term can be evaluated analytically [13], but choosing $p_\theta(\mathbf{z})$ to be the isotropic normal distribution with perfectly uncorrelated components forces the model to learn representations which encode statistically independent features about the data separately, in different positions of the code. Varying the value for β regulates the amount of the applied learning pressure and in the next section we closely examine the effect of varying this disentanglement parameter.

3 Experiments

3.1 Disentanglement level with respect to β

3.1.1 Data

Higgins *et al.* [9] made the key assumption that the observed data should possess transform continuities in order to be able to find some regularity in it in an unsupervised manner. We assume the input data is generated by factors of variation, densely sampled from their respective continuous distributions. In accordance to this considerations, we have constructed a synthetic dataset of 64x64 binarised images containing each a single shape. The generative factors defining each image are: a shape – square (□), ellipse (⬭) or triangle (△); position X (16 values); position Y (16 values); scale (6 levels); rotation (60 values over the $[0, \pi]$ range). The images were randomly separated in training, validation and test sets in a ratio 70:15:15 in a stratified way. Special care was taken to reduce the leakage between the subsets by removing duplicate images incidentally caused by some idempotent transformations (e.g. rotation of a square in $90°$, $180°$ or $270°$ produces the same figure). The final dataset consists of 267,021 images[1].

3.1.2 Measuring disentanglement

The disentanglement of an autoencoder cannot be usefully measured by its reconstruction accuracy or the KL-divergence term of the loss function as they fail to convey the notion of independence we want to obtain for the elements of the code. Precisely, disentanglement effect would mean distinguishing the generating factors of the data and encoding them in separate code elements.

Higgins *et al.* [9] proposed a disentanglement measuring method which tries to evaluate this property of the trained autoencoders. A random set of generating factors is taken, the image img_1 is constructed, and the code means $\mathbf{z}_1^\mu = encoder(img_1)$ are extracted. The same procedure is repeated, but this time one of the factors is randomly modified while all the others are kept the same. Denote the newly extracted code means with $\mathbf{z}_2^\mu$. A low capacity linear classifier is trained to map $\frac{|\mathbf{z}_1^\mu - \mathbf{z}_2^\mu|}{\max(|\mathbf{z}_1^\mu - \mathbf{z}_2^\mu|)}$ (division intended for normalisation) to the single factor that was changed during the

[1] The source code to reproduce all of our experiments described in this work can be found at `www.github.com/mpeychev/disentangled-autoencoders`.

process of obtaining $\mathbf{z}_1^\mu$ and $\mathbf{z}_2^\mu$. The classifier accuracy is then reported as a disentanglement measure of the autoencoder of interest. The assumption is that if a simple classifier is capable of inferring the single input generating factor responsible for the code perturbation, then the model provides some form of transparency and interpretability.

An alternative method in which one of the factors is fixed while all the others are randomly sampled between the generation of img_1 and img_2 is presented in a subsequent work by the same authors [10] but we only evaluate the first approach here.

3.1.3 Results

The disentanglement levels of four types of autoencoders we trained, varying β from 0 to 5 with a step of 0.2 are presented in Figure 1. Simple and denoising variants of autoencoders were considered and both fully connected and convolutional architectures were tested. The applied noise in the denoising case was salt-and-pepper, randomly flipping up to 20% of the pixels. For each β, 5 autoencoder models were trained. In all graphs here and below we plot the means of the results obtained for all models trained with the same β while the error bars denote standard deviations.

The first thing to become clear is the high variance between separate runs with the same value for β. A potential reason for that might be the method not being completely capable of closing the gap between the notion of disentanglement and factor independence we have with the underlying properties of the representations learnt by the autoencoders. For example, it was observed that for the position latents in the case of $\beta = 4$ (which is supposed to be the disentangled case according to Higgins *et al.* [9]), the autoencoder may sometimes learn "curved" or rotated, but still orthogonal, coordinate systems, which differs from what we would expect. Moreover, when reporting their results in [9], the bottom 50% of the obtained measurements have been discarded for unknown reasons (this was not performed when organising our results in Figure 1). Higgins *et al.* [9, 10] report results for fixed values of β ($\beta = 0, 1,$ and 4) only, so to the best of our knowledge the findings about the intermediate values, presented in Figure 1 constitute previously unpublished work.

Another trend is the increase in the disentanglement with bigger values for β. The growth seems to be the most steady for convolutional denoising autoencoders. This is consistent with the claims that convolutional networks might be better at capturing image structures than fully connected ones and that adding noise and reconstructing the original data could act as a good regulariser.

The assumption for bigger values of β is that at some point the autoencoder disentanglement will get flat (as starting to happen for the fully connected denoising case) and from then onwards further increase of β will be damaging, as it will come at the cost of reducing the autoencoder's reconstruction ability. This in turn can lead to losing some useful learnt properties about the data. An application in which even small (but nonzero) values of β can be harmful is described in the next section.

3.2 MNIST classification with disentangled autoencoders

After evaluating disentangled autoencoders' behaviour on our synthetic dataset, it was a natural continuation to test them against an established machine learning benchmark. As such, the MNIST [15] dataset was considered to be a suitable candidate. The evaluation procedure began with an unsupervised autoencoder training first. Subsequently, a Support Vector Machine classifier was trained (using the same training dataset) to map the image codes, produced by the encoder network, to the respective image classes. The results are presented in Figure 2. The outcomes of the experiments using 10 and 30% of the MNIST training dataset were included because the 20% ones were outlying.

Increasing the number of training examples consistently increases the classification accuracy, as expected – providing more labels helps the model generalise. Although with higher variance, the convolutional architecture seem to be more robust to training the autoencoder with fewer datapoints. The major spikes in the classification accuracy from $\beta = 0$ to $\beta = 0.1$ can be attributed to overfitting. When $\beta = 0$, the network tends to learn a one-to-one mapping and the latents may end up unrelated to their nearby values. The KL term acts as a regulariser, adding smoothness to the learnt latent manifold.

Taking into account the increasing error in the autoencoders reconstruction precision, it can be concluded that the autoencoder disentanglement is deteriorating for classification problems when applied to the MNIST dataset. This is an expected result, especially because of the lack of explicit

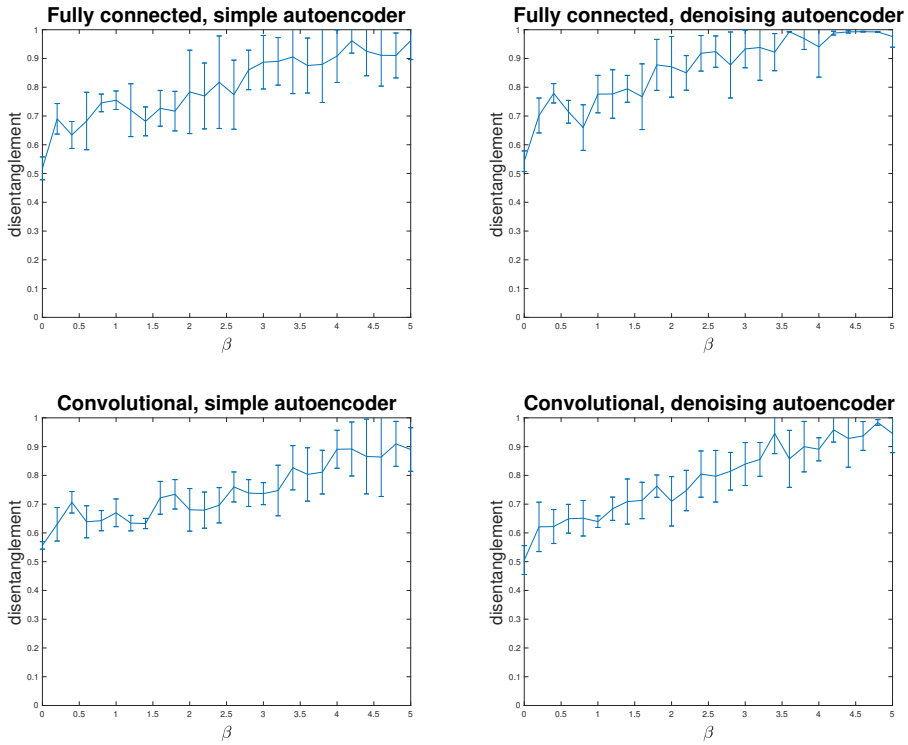


Figure 1: Disentanglement levels of the autoencoders, trained on the synthetic dataset, with respect to the parameter β.

continuity and generating factors of the MNIST images. However, it establishes the fact that there is a trade-off between the two terms of the disentangled autoencoder loss function and that they force the model to learn different properties about the data. When training a disentangled autoencoder, this trade-off should be considered and a balanced solution is desirable.

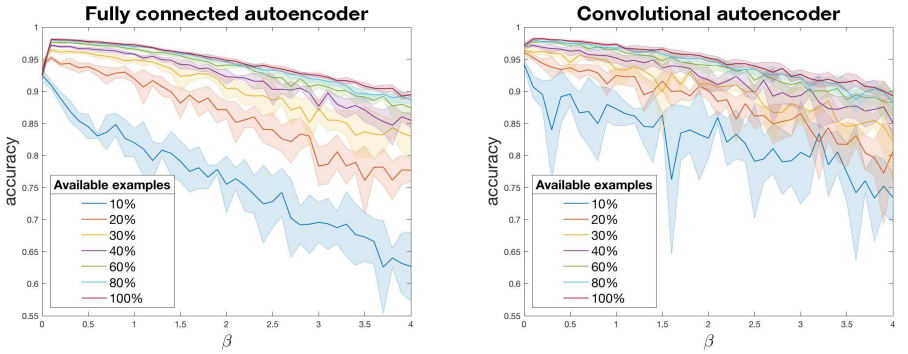


Figure 2: Results of MNIST classification with autoencoders. β goes from 0 to 4 with step 0.1. We execute the same experiments with different number of labels available during the training.[2]

4 Conclusion

This work contributes with, to the best of our knowledge, new and unpublished findings about the properties of disentangled autoencoders. In particular, their level of disentanglement was measured over a whole range of values β and it was discovered that, as expected, the disentanglement typically makes the models' performance worse in classification tasks.

[2]Best viewed in colour.

References

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[2] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59(4):291–294, 1988.

[3] Brian Cheung, J. A. Livezey, A. K. Bansal, and B. A. Olshausen. Discovering hidden factors of variation in deep networks. *CoRR*, abs/1412.6583, 2014.

[4] Taco S. Cohen and Max Welling. Transformation properties of learned visual representations. *CoRR*, abs/1412.7659, 2014.

[5] G. Desjardins, A. Courville, and Y. Bengio. Disentangling Factors of Variation via Generative Entangling. *arXiv*, October 2012.

[6] Ian Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[7] Alex Graves, A.-R. Mohamed, and G. E. Hinton. Speech recognition with deep recurrent neural networks. *CoRR*, abs/1303.5778, 2013.

[8] Kaiming He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 1026–1034, Washington, DC, USA, 2015. IEEE Computer Society.

[9] Irina Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, C. Blundell, S. Mohamed, and A. Lerchner. Early visual concept learning with unsupervised deep learning. *CoRR*, abs/1606.05579, 2016.

[10] Irina Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, C. Blundell, S. Mohamed, and A. Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2017.

[11] Geoffrey E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders. In *Proceedings of the 21th International Conference on Artificial Neural Networks - Volume Part I*, ICANN'11, pages 44–51, Berlin, Heidelberg, 2011. Springer-Verlag.

[12] Geoffrey E Hinton and R. S. Zemel. Autoencoders, minimum description length and helmholtz free energy. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 3–10. Morgan-Kaufmann, 1994.

[13] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

[14] Alex Krizhevsky, I. Sutskever, and G. E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[15] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998.

[16] Yang Li, Q. Pan, S. Wang, H. Peng, T. Yang, and E. Cambria. Disentangled variational auto-encoder for semi-supervised learning. *CoRR*, abs/1709.05047, 2017.

[17] Scott Reed, K. Sohn, Y. Zhang, and H. Lee. Learning to disentangle factors of variation with manifold interaction. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1431–II–1439. JMLR.org, 2014.

[18] Florian Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.

[19] David Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 01 2016.

[20] Christian Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.

[21] Aäron van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR*, abs/1609.03499, 2016.

Learning independent causal mechanisms

Giambattista Parascandolo[†§] gparascandolo@tue.mpg.de
Mateo Rojas-Carulla[†‡] mrojas@tue.mpg.de
Niki Kilbertus[†‡] nkilbertus@tue.mpg.de

Bernhard Schölkopf[†] bs@tue.mpg.de

[†]Max Planck Institute for Intelligent Systems
[‡]University of Cambridge
[§]Max Planck ETH Center for Learning Systems

Abstract

Independent causal mechanisms are a central concept in the study of causality with implications for machine learning tasks. In this work we develop an algorithm to recover a set of (inverse) independent mechanisms relating a distribution transformed by the mechanisms to a reference distribution. The approach is fully unsupervised and based on a set of experts that compete for data to specialize and extract the mechanisms. We test and analyze the proposed method on a series of experiments based on image transformations. Each expert successfully maps a subset of the transformed data to the original domain, and the learned mechanisms generalize to other domains. We discuss implications for domain transfer and links to recent trends in generative modeling.

1 Introduction

When presented with digits which are translated, corrupted, or inverted, humans can usually correctly label them without the need of re-learning them from scratch. The same applies for new objects, essentially after having seen them once. This may be due to the fact that human intelligence utilizes *mechanisms* (such as translation) that are generic and *generalize* across object classes. These mechanisms are *modular, re-usable and broadly applicable*, and the problem of learning them from data is a fundamental question for the study of transfer.

In the study of causality, the *independent mechanisms (IMs)* assumption states that the causal generative process of a system's variables is composed of autonomous modules that do not inform or influence each other [7, 9].[1] In the present paper, we focus on a class of such modules, and on algorithms to learn them from data. We describe an architecture using competing experts specializing on different transformations. The resulting model permits a form of lifelong learning, with the possibility of easily adding, removing, retraining, and exporting its components independently. It thus forms a minimal example of what role causality can play in addressing crucial challenges of machine learning [8].

We illustrate our approach on MNIST digits which have undergone different transformations such as contrast inversion, noise addition and translations. Information about the nature and number of such transformations need not be known at the beginning of training. Our goal is to identify the independent mechanisms linking a reference distribution to a distribution of modified digits, and

[1] See Appendix D for a more detailed description of the concept of independent mechanisms.

Workshop on Learning Disentangled Representations: from Perception to Control, 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

learn to invert them without supervision. Ultimately, such inverse mechanisms could be used to transform modified digits and classify them using a standard MNIST classifier, thus exhibiting a form of robustness that animate intelligence excels at.

Our work draws from mixtures of experts [1, 4, 10], unsupervised domain adaptation [2, 11], and causality (e.g. [6, 7]). Its novelty lies in the following aspects: (1) we automatically identify and invert a set of independent (inverse) causal mechanisms; (2) we do so using only data from an original distribution and from the mixture of transformed data, without labels; (3) the architecture is modular, can be easily expanded, and its trained modules can be reused; and (4) the method relies on competition of experts.

2 Learning causal mechanisms as independent modules

Consider a canonical distribution P on $\mathbb{R}^d$, e.g., the empirical distribution defined by MNIST digits on pixel space. We further consider N measurable functions $M_1, \dots, M_N : \mathbb{R}^d \to \mathbb{R}^d$, called *mechanisms*. We think of these as independent causal mechanisms in nature, and their number is a priori unknown. The mechanisms give rise to N distributions $Q_1, \dots, Q_N$ where $Q_j = M_j(P)$.[2] In the MNIST example, we consider translations or adding noise as mechanisms, i.e., the corresponding Q distributions are translated and noisy MNIST digits.

At training time, we receive a dataset $\mathcal{D}_Q = (x_i)_{i=1}^n$ drawn i.i.d. from a mixture of $Q_1, \dots, Q_N$, and an independent sample $\mathcal{D}_P$ from the canonical distribution P. Our goal is to identify the underlying mechanisms $M_1, \dots, M_N$ and learn approximate inverse mappings which allow us to map the examples from $\mathcal{D}_Q$ back to their counterpart drawn from P.

If we were given distinct datasets $\mathcal{D}_{Q_j}$ each drawn from Q_j, we could individually learn an independent approximation for each mechanism. In our case, we do not have access to the distinct datasets. Instead we construct a larger set $\mathcal{D}_Q$ by first taking the union of the sets D_{Q_j}, and then applying a random permutation. This corresponds to a dataset where each element has been generated by one of the (independent) mechanisms, but we do not know by which one. Clearly, it should be harder to identify and learn independent mechanisms from such a dataset. This is the setting we address below, and the crucial idea will be that of *competition*.

Competitive learning of independent mechanisms The training machine is composed of N' parametric functions $E_1, \dots, E_{N'}$ with distinct trainable parameters $\theta_1, \dots, \theta_{N'}$. We refer to these functions as the *experts*. Note that we do not require $N' = N$, since the real number of mechanisms is unknown a priori. The goal is to maximize an objective function $c : \mathbb{R}^d \to \mathbb{R}$ with the key property that c takes high values on the support of the canonical distribution P, and low values outside. Note that it is possible for c to be a parametric function, and for these parameters to be jointly optimized with the experts during training.

During training, the experts compete for the data points. Each example x' from $\mathcal{D}_Q$ is fed to all experts independently and in parallel. Depending on the output of each expert $c_j = c(E_j(x'))$, we select the winning expert E_{j^*}, where $j^* = \arg\max_j(c_j)$. E_{j^*} *wins* the example x', and its parameters θ_{j^*} are updated as to maximize $c(E_{j^*}(x'))$, while the other experts remain unchanged. The motivation behind competitively updating only the winning expert is to enforce specialization; the best performing expert becomes even better at mapping x' back to the corresponding sample from the canonical distribution. Figure 1 depicts this procedure. Overall, our optimization problem reads

$$\theta_1^*, \dots, \theta_{N'}^* = \arg\max_{\theta_1, \dots, \theta_{N'}} \mathbb{E}_{x' \sim Q} \left(\max_{j \in \{1, \dots, N'\}} c(E_{\theta_j}(x')) \right). \tag{1}$$

The training described above raises a number of questions:

1) **Convergence criterion.** Since the problem is fully unsupervised, there is no straightforward way of measuring convergence. In this work we fix the maximum number of iterations.

[2] Each distribution Q_j is defined as the pushforward measure induced by P via M_j.

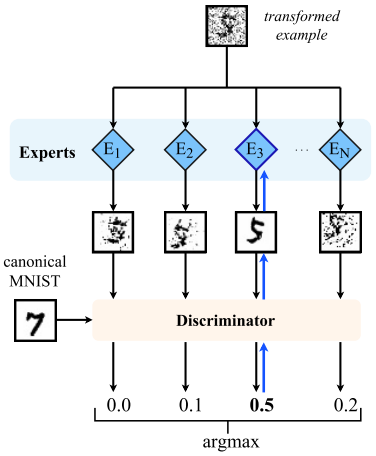


Figure 1: Only Expert 3 is specializing on denoising, it wins the example and gets trained on it, whereas the others perform translations and are not updated.

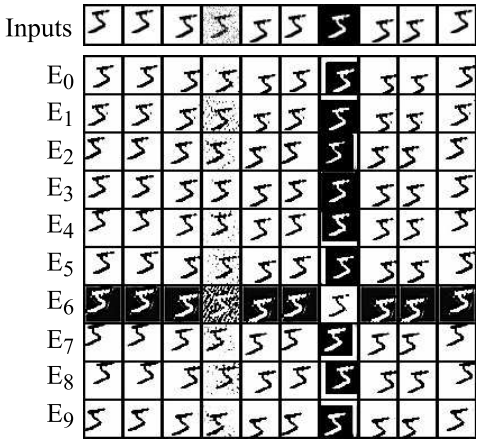


Figure 2: Each expert transforms the input presented on the top row. The experts learned the mechanisms, and consistently apply them to digits outside of their training domain.

2) **Selecting the appropriate number of experts.** Generally, the number of mechanisms N which generated the dataset $\mathcal{D}_Q$ is not available a priori, but for the experiments in this work we assume we know N. See Appendix A.3 for experiments and considerations for the cases $N' \neq N$.
3) **Time and space complexity.** Each example can be evaluated by all experts in parallel, and therefore the time complexity is bounded by that of a single expert. In principle each expert has a smaller architecture than a single large network, therefore the committee of experts will typically be faster to execute.

Concrete protocols for neural networks. One possible model class for the experts are deep neural networks. Recent advances in generative modeling give rise to natural choices for the loss function c: for instance, c could be the reconstruction loss of an autoencoder trained on the original data, or the output of a discriminator trained with adversarial training [3]. In the next section we introduce a formal description of a training procedure based on adversarial training in Algorithm 1, and present experimental evidence of its good performance.

3 Experiments

In this set of experiments we test the method presented in Section 2 on the MNIST dataset transformed with the set of mechanisms composed of eight directions of translations by 4 pixels (up, down, left, right, and the four diagonals), contrast inversion, addition of noise[3]. We split the training partition of MNIST in half, and transform all and only the examples in the first half: this ensures that there is no matching ground truth for the experts to learn the mechanisms, and that learning is fully unsupervised. Each expert E_i can be seen as a generator from a GAN, that is conditioned on an input image instead of a noise vector. A discriminator D provides gradients for training the experts and acts also as a selection mechanism c: only the expert whose output obtains the higher score from D *wins* the example, and is trained on it to maximize the output of D. We describe the exact algorithm used to train the networks in these experiments in Algorithm 1.

Neural nets details Each expert is a fully convolutional network with five layers, 32 filters per layer of size 3×3, ELU, batch normalization, and zero padding. The discriminator is also a CNN, with average pooling every two convolutional layers, growing number of filters, and a fully connected layer with 1024 neurons as last hidden layer. Both networks are trained using Adam as optimizer, with the default hyper-parameters.[4] We first pretrain each expert for up to 200 iterations on predicting

[3] See Appendix C for further details about the transformations

[4] For the exact experimental parameters and architectures see the Appendix B or the PyTorch implementation that will be released online.

Algorithm 1 Learning independent mechanisms using competition of experts and advers. training

Precondition: X: data sampled from P; X': data sampled from $\mathcal{D}_Q$; D discriminator; N': number of experts; T: maximum number of iterations;
(p) highlights that the steps in the instruction can be executed in parallel

$\{E_i \leftarrow \text{TrainNewAutoencoderOn}(X')\}_{j=1}^{N'}$ ▷ Init set of experts as approx identity **(p)**
for $t \leftarrow 1$ to T **do**
  $x, x' \leftarrow \text{Sample}(X), \text{Sample}(X')$ ▷ Sample minibatches
  $\{c_j \leftarrow D(E_j(x'))\}_{j=1}^{N'}$ ▷ Scores from D for all outputs from the experts **(p)**
  $\theta_D^{t+1} \leftarrow \text{Adam}(\theta_D^t, \nabla \log D(x) + \nabla(1/N' \sum_{j=1}^{N'} \log(1 - c_j)))$ ▷ Update D **(p)**
  $\{\theta_{E_j}^t \leftarrow \text{Adam}(\theta_{E_j}^t, \nabla \max_{j \in 1,\ldots,N'} \log(c_j))\}_{j=1}^{N'}$ ▷ Update experts **(p)**

identical input-output pairs randomly selected from the transformed dataset, which improved the speed and robustness of convergence (see Appendix A.2). We run the experiments 10 times with different random seeds for the initializations. Each experiment is run for 2000 iterations.

4 Results

The experts correctly specialized on inverting exactly one mechanism each in 7 out of the 10 runs; in the remaining 3 runs the results were only slightly suboptimal: one expert specialized on two tasks, one expert did not specialize on any, and the remaining experts specialized on one task each, thus covering all the existing tasks. In Figure 3 we show a randomly selected batch of inputs and corresponding outputs from the model. Each independent mechanism was inverted by a different expert.

Figure 3: The top row contains 16 random inputs to the networks, and the bottom row the corresponding outputs from the highest scoring experts against the discriminator after 1000 iterations.

1. The experts specialize w.r.t. c . We encourage the reader to look at Figure 5 in Appendix A.1, which shows how after an initial chaotic phase of heavy competition, the experts exhibit the desired behavior and obtain a high score on D on one mechanism each. Also Figure 6 provides further evidence, by visualizing that the clusters induced by c are meaningful.

2. The transformed outputs improve a classifier. We also test the output of our experts on the test partition of the data against a pretrained MNIST classifier. We compute the accuracy for three inputs: *a*) the transformed test digits, *b*) the transformed digits after being processed by the highest scoring experts, *c*) the original test digits. The latter can be seen as an upper bound to the accuracy that can be achieved. As shown by the two dashed horizontal lines in Figure 4, the transformed test digits achieve a 40% accuracy when tested directly on the classifier, while the untransformed digits would achieve $\approx 99\%$ accuracy. The accuracy for the output digits also starts at 40% — due to the identity initialization — and quickly matches the performance of the original digits as it is trained. Note also that after about 600 iterations — i.e. as the networks have seen overall about one third of the whole dataset, and once only — the accuracy is already almost at the upper bound.

3. The experts learn mechanisms. Finally, we test the networks on inputs that were not transformed with the mechanisms that each of them has learned to invert. As shown in Figure 2, each network consistently applies the same transformation also on inputs outside of its training distribution, and therefore the experts not only recovered the correct digits for the domain they have specialized on, but indeed learned the independent *mechanisms*. Since the experts are fully convolutional networks in this experiment, they could be even be ported to other domains with images of different sizes.

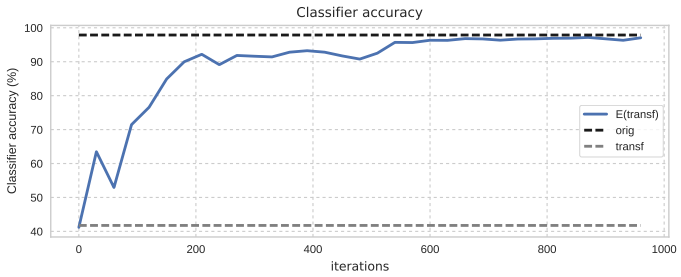


Figure 4: Accuracy on the transformed test digits $\mathcal{D}_Q$ of a pretrained CNN MNIST classifier, on the same digits after going through our model, and on the original digits before transformation $\mathcal{D}_P$ (here $\mathcal{D}_P$ corresponds to the ground truth pre-image of the digits in $\mathcal{D}_Q$ for the true applied mechanisms).

In Appendix A.2 we describe how a baseline based on a single network fails at learning multiple mechanisms in the same setting.

5 Conclusions

We have developed a method to identify and learn a set of independent causal mechanisms, reporting promising results in an experiment based on image transformations. Future work could explore more complex settings and diverse domains. Another interesting direction consists in independent mechanisms that *simultaneously* affect the data, for which one could allow multiple passes through the committee of experts to identify local mechanisms (akin to Lie derivatives) — for instance, using recurrent neural nets.

We believe our work constitutes a relevant connection between causal modeling and deep learning. As discussed in the introduction and Appendix D, causality has a lot to offer for crucial machine learning problems such as transfer or compositional modeling. Our systems illustrates some of these properties. Independent modules as sub-components could be trained independently and/or from multiple domains, added subsequently, and transferred to other problems. This may constitute a step towards causally motivated life-long learning.

References

[1] R. Aljundi, P. Chakravarty, and T. Tuytelaars. Expert gate: Lifelong learning with a network of experts. *arXiv preprint arXiv:1611.06194*, 2016.

[2] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *arXiv preprint arXiv:1612.05424*, 2016.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[4] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[5] D. Janzing and B. Schölkopf. Causal inference using the algorithmic Markov condition. *IEEE Transactions on Information Theory*, 56(10):5168–5194, 2010.

[6] J. Pearl. *Causality*. Cambridge University Press, 2000.

[7] J. Peters, D. Janzing, and B. Schölkopf. *Elements of Causal Inference*. The MIT Press, 2017.

[8] B. Schölkopf, D. Janzing, and D. Lopez-Paz. Causal and statistical learning. In A. Christmann, K. Jetter, S. Smale, and D.-X. Zhou, editors, *Oberwolfach Reports*, volume 33, pages 1896–1899, 2016.

[9] B. Schölkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. M. Mooij. On causal and anticausal learning. In J. Langford and J. Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 1255–1262, New York, NY, USA, 2012. Omnipress.

[10] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[11] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*, 2017.

A Additional results.

A.1 Plot of competing experts and cluster assignments.

Each expert in Figure 5 is represented with the same color and linestyle across all tasks. Note how the red expert tries to learn two similar tasks until iteration 500 (i.e. left and up-left translation), when the green expert takes over one of the tasks and they can then both quickly specialize.

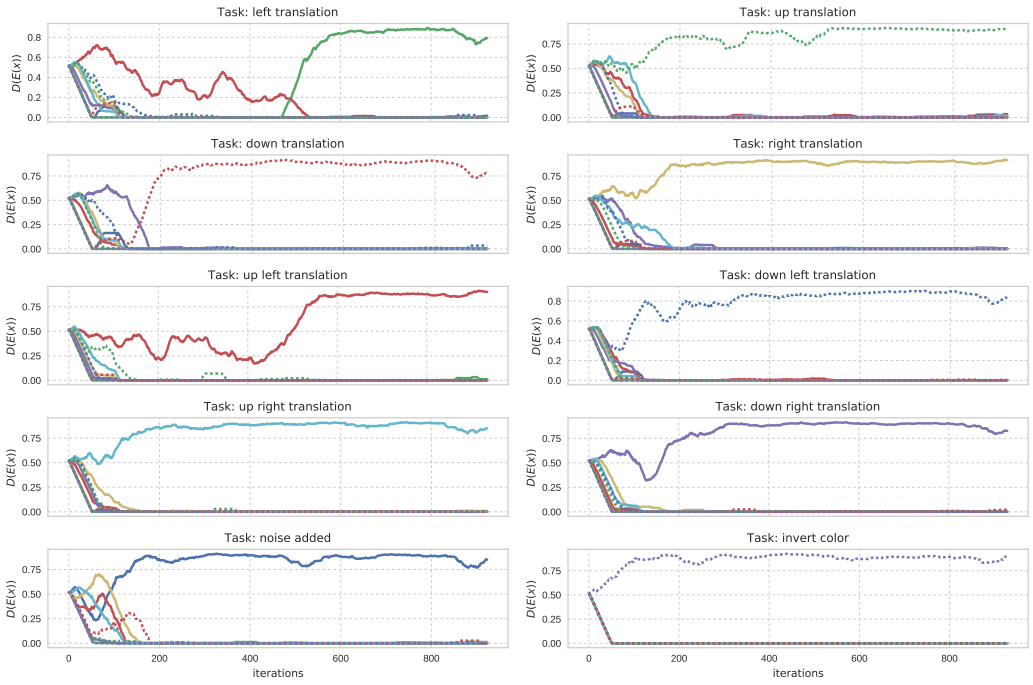


Figure 5: Each line style is associated to the score that an expert obtains on the discriminator when being fed transformed digits using one of the mechanisms. Each expert learns to specialize on a different mechanism. Each curve is smoothed with an average of the last 50 iterations for ease of visualization.

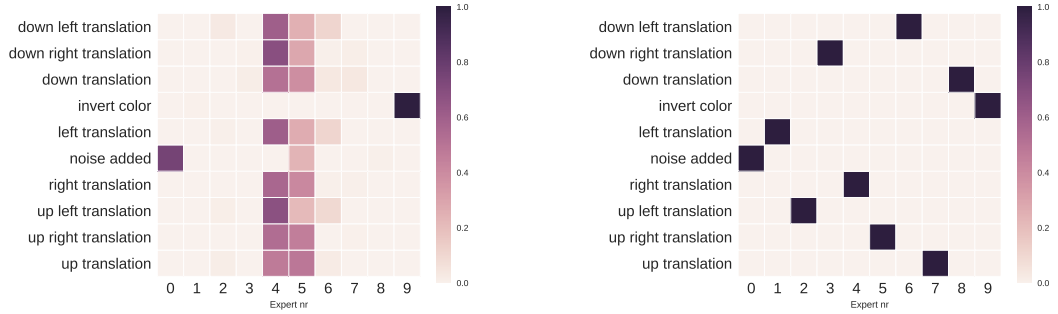
A.2

Effect of the approximate identity initialization When running the same experiments without the approximate identity initialization, we found that often several experts fail to specialize. Out of 10 new runs with random initialization, only one experiment had arguably good results, with eight experts specializing on one task each, one expert on two tasks, and the last expert on none. The performance was worse in the remaining runs. We tested whether the problem was that the algorithm takes longer to converge following a random initialization, and ran one additional experiment for $10,000$ iterations. The results did not improve.

A simple single-net baseline Training a single network instead of a committee of experts, once with 32, once with 64, and once with 128 filters per layer, it did not learn more than one inverse mechanism. Note that a single network with 128 filters per layer has slightly more parameters overall than the committee of 10 experts with 32 filters per layer each. While careful hyperparameter tuning may allow us to learn more mechanisms, it seems at least not to be straightforward.

A.3 Too many or too few experts.

Too many experts When there are too many experts, for most tasks only one wins all the examples, as shown in Figure 7 where the model has 16 experts for 10 tasks. The remaining experts either do not specialize at all — and therefore can be removed from the architecture — or specialize on the same task, and could therefore be combined if after inspection they are considered to perform the same task. Since the accuracy on the transformed data tested on the pretrained classifier reaches again the upperbound of the untransformed data, and since the progress is very similar to that illustrated in Figure 4, we omit this plot.



(a) At the beginning of training

(b) After 1000 iterations

Figure 6: The proportion of data won by each expert for each transformation on the digits from the test set.

Too few experts For a committee of 6 experts, the networks do not reconstruct properly most of the digits, which is reflected by an overall low objective function value on the data. Also, the score against the classifier that does not exceed 72%. A few experts are inevitably assigned to multiple tasks, and by looking at Figure 7 it is interesting to see that the clustering result is still meaningful (e.g. expert 5 is assigned to left, down-left, and up-left translation).

B Details of neural networks

In Table 1 we report the configuration of the neural networks used in these experiments.

C Transformations

In our experiments we use the following transformations

- Translations: the image is shifted by 4 pixels in one of the eight directions up, down, left, right and the four diagonals.
- Color inversion: (or contrast inversion) the value of each pixel — originally in the range $[0, 1]$ — is recomputed as $1-$ the original value.
- Noise addition: random Gaussian noise with zero mean and variance 0.25 is added to the original image, which is then clamped again to the $[0, 1]$ interval.

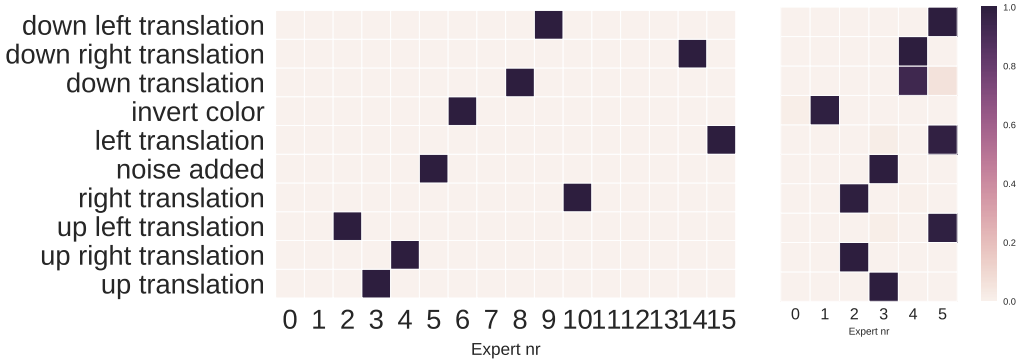


Figure 7: The proportion of data won by each expert for each transformation on the digits from the test set, for the case of 10 mechanisms and more experts (16 on left) or too few (6 on the right).

Table 1: Architectures of the neural networks used in the experiment section. BN stands for Batch normalization, FC for fully connected. All convolutions are preceded by a 1 pixel zero padding.

Expert
Layers
$3 \times 3, 32$, BN, ELU
$3 \times 3, 32$, BN, ELU
$3 \times 3, 32$, BN, ELU
$3 \times 3, 32$, BN, ELU
$3 \times 3, 1$, sigmoid

Discriminator
Layers
$3 \times 3, 16$, ELU
$3 \times 3, 16$, ELU
$3 \times 3, 16$, ELU
2×2, avg pooling
$3 \times 3, 32$, ELU
$3 \times 3, 32$, ELU
2×2, avg pooling
$3 \times 3, 64$, ELU
$3 \times 3, 64$, ELU
2×2, avg pooling
1024, FC, ELU
1, FC, sigmoid

D Notes on the Formalization of Independence of Mechanisms

If a joint density is Markovian with respect to a directed graph $\mathcal{G}$, we can write it as

$$p(\mathbf{x}) = p(x_1, \ldots, x_d) = \prod_{j=1}^{d} p(x_j | \text{pa}_{\mathcal{G}}^j), \tag{2}$$

where $\text{pa}_{\mathcal{G}}^j$ denotes the parents of variable x_j in the graph.

For a given joint density, there are usually many decompositions of the form (2), with respect to different graphs. If we use the *causal* graph, i.e., if the edges of $\mathcal{G}$ denote direct causation [6], then the conditional $p(x_j|\text{pa}_{\mathcal{G}}{}^j)$ can be thought of as physical mechanism generating x_j from its parents (sometimes referred to as a *causal conditional*). In this case, we can think of (2) as a *generative* model where the term "generative" truly refers to a physical generative process. In the alternative view of causal models as structural equation models, each of the causal conditionals corresponds to a functional mapping and a noise variable [6].

By the IM assumption, the causal conditionals are autonomous modules that do not influence or inform each other. This has multiple consequences. First, knowledge of one mechanism does not contain information about another one (Section D.1). Second, if one mechanism changes (e.g., due to distribution shift), there is no reason that other mechanisms should also change, i.e., they tend to remain *invariant*. As a special case, it is (in principle) possible to locally *intervene* on one mechanism (for instance, by setting it to a constant) without affecting any of the other modules.

The IM assumption can be exploited when performing causal structure inference [7]. However, it also has implications for machine learning more broadly. A model which is expressed in terms of causal conditionals (rather than conditionals with respect to some other factorization) is likely to have components that better transfer or generalize to other settings [9], and its modules are better suited for building complex models from simpler ones. Independent modules as sub-components can be trained independently, from multiple domains, are more likely to be re-usable. They can also be easier to interpret since they correspond to physical mechanisms. Animate intelligence cannot afford to learn new models from scratch for every new task. Rather, it is likely to rely on robust local components that can flexibly be re-used and re-purposed. It also requires local mechanisms for adapting and training modules rather than re-training the whole brain every time a new task is learned.

D.1 Algorithmic complexity

In this section we briefly discuss the notion of independence of mechanisms as in [5], where the independence principle is formalized in terms of algorithmic complexity (also known as Kolmogorov complexity). We summarize the main points needed in the present context. We parametrize each *mechanism* by a bit string x. The Kolmogorov complexity $K(x)$ of x is the length of the shortest program generating x on an a priori chosen universal Turing machine. The **algorithmic mutual information** can be defined as $I(x : y) := K(x) + K(y) - K(x, y)$, and it can be shown to equal

$$I(x : y) = K(y) - K(y|x^*), \tag{3}$$

where for technical reasons we need to work with x^*, the shortest description of x (which is in general uncomputable). Here, the conditional Kolmogorov complexity $K(y|x)$ is defined as the length of the shortest

program that generates y from x. The algorithmic mutual information measures the algorithmic information two objects have in common. We define two mechanisms to be **(algorithmically) independent** whenever the length of the shortest description of the two bit strings together is not shorter than the sum of the shortest individual descriptions (note it cannot be longer), i.e., if their algorithmic mutual information vanishes.[5] In view of (3), this means that

$$K(y) = K(y|x^*). \tag{4}$$

We will say that two mechanisms x and y are independent whenever the complexity of the conditional mechanism $y|x$ is comparable to the complexity of the unconditional one y. If, in contrast, the two mechanisms were closely related, then we would expect that we can mimick one of the mechanisms by applying the other one followed by a low complexity conditional mechanism.

This can be implemented by having a complexity measure for, say, neural networks, and comparing the complexities of neural nets that are trained to perform certain tasks. Inspired by regularization theory, we could measure complexity by inverse regularization strength. A simple way to regularize neural nets consists of early stopping. If we fix the number of training epochs to a constant, and find that network 1 reaches a lower error than network 2, we conclude that the network 2 would take longer to reach the same low error, and thus network 2 requires higher effective complexity to solve its task than network 1. We have run preliminary experiments with this measure and found that (1) indeed our training procedure did increase independence, and (2) the independence between two different mechanism was larger than the independence between one mechanism and the identity.

[5] All statements are valid up to additive constants, linked to the choice of a Turing machine which produces the object (bit string) when given its compression as an input. For details, see [5].

On Decomposing Motion and Content for Video Generation

Sergey Tulyakov
Snap Research
stulyakov@snap.com

Ming-Yu Liu, Xiaodong Yang, Jan Kautz
NVIDIA
{mingyul,xiaodongy,jkautz}@nvidia.com

Abstract

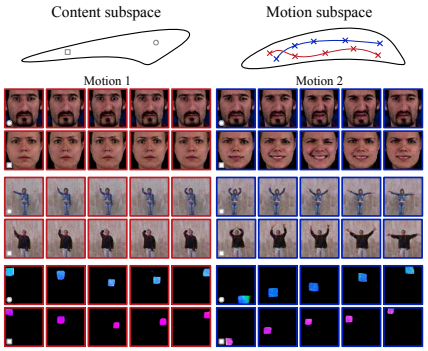
Visual information in a natural video can be decomposed into two major components: content and motion. While content encodes the objects present in the video, motion encodes the object dynamics. Based on this prior, we propose the Motion and Content decomposed Generative Adversarial Network (MoCoGAN) framework for video generation. The proposed framework generates a video clip by sequentially mapping random noise vectors to video frames. We divide a random noise vector into content and motion parts. The content part, modeled by a Gaussian, is kept fixed when generating individual frames in a short video clip, since the content in a short clip remains largely the same. On the other hand, the motion part, modeled by a recurrent neural network, aims at representing the dynamics in a video. Despite the lack of supervision signals on the motion and content decomposition in natural videos, we show that the MoCoGAN framework can learn to decompose these two factors through a novel adversarial training scheme. More information is available in our project page (https://github.com/sergeytulyakov/mocogan).

1 Introduction

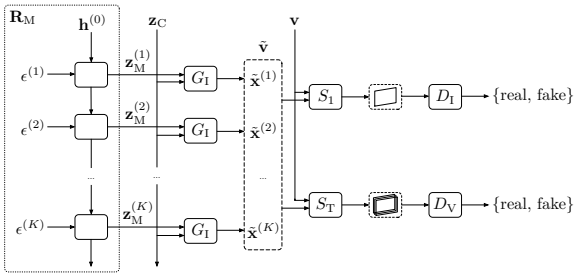
Video generation is hard for the following reasons. First, since a video is a spatio-temporal recording of visual information of objects performing various actions, a generative model needs to learn the physical motion models of objects in addition to learning their appearance models. If the learned object motion models are incorrect, the generated video may contain objects performing physically implausible motion. Second, the time dimension brings in a huge amount of variation. Just imagine the amount of speed variations that a person can have as performing a squat movement. Each speed pattern results in a different video, although the appearances of the human in the videos are the same. Third, but not last, as human beings have evolved to be rather sensitive to motion, motion artifacts are particularly pronounced to human eyes. It is more difficult to generate visually convincing videos.

Recently, a few attempts to the image generation problem were made through generative adversarial networks [5]. In [7] hypothesized that a video clip is a point in a latent space and proposed learning a mapping from the latent space to video clips. We argue that assuming a video clip is a point in the latent space unnecessarily increases the complexity of the problem, because videos of the same action with different execution speed are represented by different points in the latent space. Moreover, this assumption forces every generated video clip to have the same length, while the length of real-world video clips varies. An alternative (and likely more intuitive and efficient) approach would assume a latent space of images and consider that a video clip is generated by traversing the points in the latent space. Video clips of different lengths correspond to paths of different lengths. In addition, as videos are about objects (content) performing actions (motion), the latent space of images should be further decomposed into two subspaces, where the deviation of a point in the first subspace (the content subspace) leads to content changes in a video clip and the deviation in the second subspace (the motion subspace) results in motions. A video clip of a person performing an action will be

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.



(a) Decomposing content and motion



(b) MoCoGAN architecture for video generation

Figure 1: (a) The MoCoGAN framework adopts a motion and content decomposed representation for video generation. (b) The MoCoGAN framework for video generation. For a video, the content vector, $\mathbf{z}_\mathrm{C}$, is sampled once and fixed. Then, a series of random variables $[\epsilon^{(1)}, ..., \epsilon^{(K)}]$ is sampled and mapped to a series of motion codes $[\mathbf{z}_\mathrm{M}^{(1)}, ..., \mathbf{z}_\mathrm{M}^{(K)}]$ via the recurrent neural network R_M. A generator G_I produces a frame, $\tilde{\mathbf{x}}^{(k)}$, using the content and the motion vectors $\{\mathbf{z}_\mathrm{C}, \mathbf{z}_\mathrm{M}^{(k)}\}$. The discriminators, D_I and D_V, are trained on real and fake images and videos, respectively, sampled from the training set $\mathbf{v}$ and the generated set $\tilde{\mathbf{v}}$. The function S_1 samples a single frame from a video, S_T samples T consecutive frames.

represented by a point in the content subspace and a trajectory in the motion subspace. Through this modeling, videos of an action with different execution speeds will only result in different traversing speeds of a trajectory in the motion space.

Decomposing motion and content allows a more controlled video generation process. By changing the content representation while fixing the motion trajectory, we have videos of different objects performing the same action. By changing motion trajectories while fixing the content representation, we have videos of the same object performing different actions. Examples of such control from our experiment results are illustrated in Figure 1a.

2 Latent Space Representation

We assume a latent space of images $Z_\mathrm{I} \equiv \mathbb{R}^d$ where each point $\mathbf{z} \in Z_\mathrm{I}$ represents an image, and a video of K frames is represented by a path of length K in the latent space, $[\mathbf{z}^{(1)}, ..., \mathbf{z}^{(\mathrm{K})}]$. By adopting this formulation, videos of different lengths can be generated by paths of different lengths. Moreover, videos of the same action executed with different speeds can be generated by traversing a same path in the latent space with different speeds.

We further assume Z_I is decomposed into the content Z_C and motion Z_M subspaces: i.e., $Z_\mathrm{I} = Z_\mathrm{C} \times Z_\mathrm{M}$ where $Z_\mathrm{C} = \mathbb{R}^{d_\mathrm{C}}$, $Z_\mathrm{M} = \mathbb{R}^{d_\mathrm{M}}$, and $d = d_\mathrm{C} + d_\mathrm{M}$. The content subspace models motion-independent appearance in videos, while the motion subspace models motion-dependent appearance in videos. For example, in a video of a person making a smile, content represents the identity of the person, while motion represents the changes of facial muscle configurations of the person. A pair of the person's identity and the facial muscle configuration represents a face image of the person. A sequence of these pairs represents a video clip of the person making a smile. By swapping the look of the person with the look of another person, a video of a different person making a smile is represented.

We model the content subspace using a Gaussian distribution: $\mathbf{z}_\mathrm{C} \sim p_{Z_\mathrm{C}} \equiv \mathcal{N}(\mathbf{z}|0, I_{d_\mathrm{C}})$ where I_{d_C} is an identity matrix of size $d_\mathrm{C} \times d_\mathrm{C}$. Based on the observation that the content remains largely the same in a short video clip, we use the same realization $\mathbf{z}_\mathrm{C}$ for generating different frames in a video clip. Motion in the video clip is modeled by a path in the motion subspace Z_M. The sequence of vectors for generating a video is represented by

$$[\mathbf{z}^{(1)}, ..., \mathbf{z}^{(K)}] = \left[\begin{bmatrix} \mathbf{z}_\mathrm{C} \\ \mathbf{z}_\mathrm{M}^{(1)} \end{bmatrix}, ..., \begin{bmatrix} \mathbf{z}_\mathrm{C} \\ \mathbf{z}_\mathrm{M}^{(K)} \end{bmatrix} \right] \tag{1}$$

where $\mathbf{z}_{\mathrm{C}} \in Z_{\mathrm{C}}$ and $\mathbf{z}_{\mathrm{M}}^{(k)} \in Z_{\mathrm{M}}$ for all k's. Since not all paths in Z_{M} correspond to physically plausible motion, we need to learn to generate valid paths. We model the path generation process using a recurrent neural network.

Let R_{M} be a recurrent neural network. At each time step, it takes a vector sampled from a Gaussian distribution as input: $\epsilon^{(k)} \sim p_{\mathrm{E}} \equiv \mathcal{N}(\epsilon|0, I_{d_{\mathrm{E}}})$ and outputs a vector in Z_{M}, which is used as the motion representation. Let $R_{\mathrm{M}}(k)$ be the output of the recurrent neural network at time k. Then, $z_{\mathrm{M}}^{(k)} = R_{\mathrm{M}}(k)$. Intuitively, the function of the recurrent neural network is to map a sequence of independent and identically distributed (i.i.d.) random variables $[\epsilon^{(1)}, \epsilon^{(2)}, ..., \epsilon^{(K)}]$ to a sequence of correlated random variables $[R_{\mathrm{M}}(1), R_{\mathrm{M}}(2), ..., R_{\mathrm{M}}(K)]$ representing the dynamics in a video. We implement R_{M} using a one-layer GRU network [4].

3 Network Architecture

MoCoGAN consists of 4 sub-networks, which are the recurrent neural network R_{M}, the image generator G_{I}, the image discriminator D_{I}, and the video discriminator D_{V}. The image generator generates a video clip by sequentially mapping vectors in Z_{I} to images, from a sequence of vectors $[\begin{bmatrix}\mathbf{z}_{\mathrm{C}} \\ \mathbf{z}_{\mathrm{M}}^{(1)}\end{bmatrix}, ..., \begin{bmatrix}\mathbf{z}_{\mathrm{C}} \\ \mathbf{z}_{\mathrm{M}}^{(K)}\end{bmatrix}]$ to a sequence of images $\tilde{\mathbf{v}} = [\tilde{\mathbf{x}}^{(1)}, ..., \tilde{\mathbf{x}}^{(K)}]$ where $\tilde{\mathbf{x}}^{(k)} = G_{\mathrm{I}}(\begin{bmatrix}\mathbf{z}_{\mathrm{C}} \\ \mathbf{z}_{\mathrm{M}}^{(k)}\end{bmatrix})$ and $\mathbf{z}_{\mathrm{M}}^{(k)}$'s are from the recurrent neural network R_{M}. We note that the video length K can vary for each video generation.

Both D_{I} and D_{V} play the judge role, providing criticisms to G_{I} and R_{M}. The image discriminator D_{I} is specialized in criticizing G_{I} based on individual images. It is trained to output 1 for a video frame sampled from a real video clip $\mathbf{v}$ and 0 for a video frame sampled from $\tilde{\mathbf{v}}$. On the other hand, D_{V} provides critics to G_{I} based on the generated video clip. D_{V} takes a fixed length video clip, say T frames. It is trained to output 1 for a video clip sampled from a real video and 0 for a video clip sampled from $\tilde{\mathrm{v}}$. Different to D_{I} which is based on vanilla CNN architecture, D_{V} is based on spatio–temporal CNN architecture. We note that the clip length T is a hyperparameter, which is set to 16 throughout our experiments. We also note that T can be smaller than the generated video length K. A video of length K can be divided into $K - \mathrm{T} + 1$ clips in a sliding-window fashion, and each of the clips can be fed into D_{V}.

The video discriminator D_{V} also evaluates the generated motion. Since G_{I} has no concept of motion, the criticisms on the motion part go directly to the recurrent neural network R_{M}. For generating a video with realistic dynamics for fooling D_{V}, R_{M} has to learn to generate a sequence of motion codes $[z_M^{(1)}, ..., z_M^{(K)}]$ from a sequence of i.i.d. noise inputs $[\epsilon^{(1)}, ..., \epsilon^{(K)}]$ in a way such that G_{I} can sequentially map $z^{(k)} = [z_C,\ z_{\mathrm{M}}^{(k)}]$ to consecutive frames in a video.

Ideally, D_{V} alone should be sufficient for training G_{I} and R_{M}, because D_{V} provides feedback on both static image appearance and video dynamics. However, we found that using D_{I} significantly improves the convergence of the adversarial training. This is because training D_{I} is simpler for it only needs to focus on static appearances. Once D_{I} is well-trained, G_{I} can learn to generate realistic images based on D_{I}'s feedback.

4 Experiments

We used the following datasets in our experiments:

- **Shape motion.** The dataset contained two types of shapes (circles and squares) with varying sizes and colors, performing two types of motion: one moving from left to right, the other moving from top to bottom. The motion trajectories were sampled from Bezier curves. There were 4,000 videos in the dataset, where the image resolution was 64×64 and video length was 16.
- **Facial expression.** We used the MUG Facial Expression Database [1] for the experiment. The dataset consisted of 86 subjects. Each video consisted of 50 to 160 frames. We cropped the face regions and scaled them to 96×96 pixels. We discarded videos containing fewer than 64 frames and used only the sequences representing one of the six facial expressions: anger, fear, disgust, happiness, sadness, and surprise. Totally, the training datasets consists of 1254 video clips.
- **Human actions.** We used the Weizmann Action database [6], containing 81 videos of 9 people performing 9 actions including jumping-jack and waving-hands. We scaled the videos to have a resolution of 96×96. Due to the small size, we did not conduct quantitative evaluation using the dataset. Instead, we provided visualization results.

(a) Shape motion

(b) Facial expressions

(c) Human actions

(d) Tai-Chi

Figure 2: The figure is best viewed via the Acrobat Reader. Click the image to play the video clip.

Table 1: Comparison with VGAN

ACD	Shape Motion	Facial Expressions
Reference	0	0.116
VGAN [7]	5.02	0.322
MoCoGAN	**1.79**	**0.201**

- **Tai-Chi.** We downloaded 4500 Tai Chi video clips from YouTube. For each clip, we applied human pose estimator [3] and cropped the clip so that the performer is in the center. Videos were scaled to 64×64 pixels.

For quantitative comparison, we measured content consistency of a generated video using the Average Content Distance (ACD) metric. For shape motion, we first computed the average color of the generated shape in each frame. Each frame was then represented by a 3-dimensional vector (RGB values). The ACD is then given by the average pairwise L2 distance of the per-frame average color vectors. For facial expression videos, we employed OpenFace [2], which outperforms human performance in the face recognition task, for measuring video content consistency. OpenFace produced a feature vector for each frame in a face video. The ACD was then computed using the average pairwise L2 distance of the per-frame feature vectors. We observed that MoCoGAN outperformed VGAN on both tasks.

In Figure 2, we visualized video generation results on the shape motion, facial expression, human actions and Tai-Chi datasets. We noted that the proposed framework was able to generate realistically looking videos on all the tasks. Moreover the framework was trained to generate 16 frames only, however, during testing time is able to realistically looking animations containing more than 16 frames.

References

[1] Niki Aifanti, Christos Papachristou, and Anastasios Delopoulos. The mug facial expression database. In *Image Analysis for Multimedia Interactive Services (WIAMIS), 2010 11th International Workshop on*, pages 1–4. IEEE, 2010. 3

[2] Brandon Amos, Bartosz Ludwiczuk, and Mahadev Satyanarayanan. Openface: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016. 4

[3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Computer Vision and Patter Recognition*, 2017. 4

[4] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 3

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014. 1

[6] Lena Gorelick, Moshe Blank, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. *PAMI*, 29(12):2247–2253, 2007. 3

[7] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, 2016. 1, 4

Understanding disentangling in β-VAE

Christopher P. Burgess, Irina Higgins, Arka Pal,
Loic Matthey, Nick Watters, Guillaume Desjardins, Alexander Lerchner
DeepMind
London, UK
{cpburgess,irinah,arkap,lmatthey,nwatters,gdesjardins,lerchner}@google.com

Abstract

We present new intuitions and theoretical assessments of the emergence of disentangled representation in variational autoencoders. Taking a rate-distortion theory perspective, we show the circumstances under which representations aligned with the underlying generative factors of variation of data emerge when optimising the modified ELBO bound in β-VAE, as training progresses. From these insights, we propose a modification to the training regime of β-VAE, that progressively increases the information capacity of the latent code during training. This modification facilitates the robust learning of disentangled representations in β-VAE, without the previous trade-off in reconstruction accuracy.

1 Introduction

Representation learning lies at the core of machine learning research. From the hand-crafted feature engineering prevalent in the past [10] to implicit representation learning of the modern deep learning approaches [22, 13, 38], it is a common theme that the performance of algorithms is critically dependent on the nature of their input representations. Despite the recent successes of the deep learning approaches [13, 38, 12, 30, 31, 29, 28, 18, 37], they are still far from the generality and robustness of biological intelligence [24]. Hence, the implicit representations learnt by these approaches through supervised or reward-based signals appear to overfit to the training task and lack the properties necessary for knowledge transfer and generalisation outside of the training data distribution.

Different ways to overcome these shortcomings have been proposed in the past, such as auxiliary tasks [18] and data augmentation [41]. Another less explored but potentially more promising approach might be to use task-agnostic unsupervised learning to learn features that capture properties necessary for good performance on a variety of tasks [3, 25]. In particular, it has been argued that disentangled representations might be helpful [3, 34].

A disentangled representation can be defined as one where single latent units are sensitive to changes in single generative factors, while being relatively invariant to changes in other factors [3]. For example, a model trained on a dataset of 3D objects might learn independent latent units sensitive to single independent data generative factors, such as object identity, position, scale, lighting or colour, similar to an inverse graphics model [23]. A disentangled representation is therefore factorised and often interpretable, whereby different independent latent units learn to encode different independent ground-truth generative factors of variation in the data.

Most initial attempts to learn disentangled representations required supervised knowledge of the data generative factors [17, 35, 32, 44, 43, 11, 23, 6, 42, 19]. This, however, is unrealistic in most real world scenarios. A number of purely unsupervised approaches to disentangled factor learning have been proposed [36, 9, 39, 7, 8, 5, 14], including β-VAE [14], the focus of this text.

β-VAE is a state of the art model for unsupervised visual disentangled representation learning. It is a modification of the Variational Autoencoder (VAE) [21, 33] objective, a generative approach that

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

aims to learn the joint distribution of images $\mathbf{x}$ and their latent generative factors $\mathbf{z}$. β-VAE adds an extra hyperparameter β to the VAE objective, which constricts the effective encoding capacity of the latent bottleneck and encourages the latent representation to be more factorised. The disentangled representations learnt by β-VAE have been shown to be important for learning a hierarchy of abstract visual concepts conducive of imagination [16] and for improving transfer performance of reinforcement learning policies, including simulation to reality transfer in robotics [15]. Given the promising results demonstrating the general usefulness of disentangled representations, it is desirable to get a better theoretical understanding of how β-VAE works as it may help to scale disentangled factor learning to more complex datasets. In particular, it is currently unknown what causes the factorised representations learnt by β-VAE to be axis aligned with the human intuition of the data generative factors compared to the standard VAE [21, 33]. Furthermore, β-VAE has other limitations, such as worse reconstruction fidelity compared to the standard VAE. This is caused by a trade-off introduced by the modified training objective that punishes reconstruction quality in order to encourage disentanglement within the latent representations. This paper attempts to shed light on the question of why β-VAE disentangles, and to use the new insights to suggest practical improvements to the β-VAE framework to overcome the reconstruction-disentanglement trade-off.

We first discuss the VAE and β-VAE frameworks in more detail, before introducing our insights into why reducing the capacity of the information bottleneck using the β hyperparameter in the β-VAE objective might be conducive to learning disentangled representations. We then propose an extension to β-VAE motivated by these insights that involves relaxing the information bottleneck during training enabling it to achieve more robust disentangling and better reconstruction accuracy.

2 Variational Autoencoder (VAE)

Suppose we have a dataset $\mathbf{x}$ of samples from a distribution parametrised by ground truth generative factors $\mathbf{z}$. The variational autoencoder (VAE) [21, 33] aims to learn the marginal likelihood of the data in such a generative process:

$$\max_{\phi,\theta} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] \tag{1}$$

where ϕ, θ parametrise the distributions of the VAE encoder and the decoder respectively. This can be re-written as:

$$\log p_\theta(\mathbf{x}|\mathbf{z}) = D_{KL}\big(q(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\big) + \mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z}) \tag{2}$$

where $D_{KL}\big(\parallel\big)$ stands for the non-negative Kullback–Leibler divergence between the true and the approximate posterior. Hence, maximising $\mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z})$ is equivalent to maximising the lower bound to the true objective in Eq. 1:

$$\log p_\theta(\mathbf{x}|\mathbf{z}) \geq \mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\big) \tag{3}$$

In order to make the optimisation of the objective in Eq. 3 tractable in practice, assumptions are commonly made. The prior $p(\mathbf{z})$ and posterior $q_\phi(\mathbf{z}|\mathbf{x})$ distributions are parametrised as Gaussians with a diagonal covariance matrix; the prior is typically set to the isotropic unit Gaussian $\mathcal{N}(0, 1)$. Parametrising the distributions in this way allows for use of the "reparametrisation trick" to estimate gradients of the lower bound with respect to the parameters ϕ, where each random variable $z_i \sim q_\phi(z_i|\mathbf{x}) = \mathcal{N}(\mu_i, \sigma_i)$ is parametrised as a differentiable transformation of a noise variable $\epsilon \sim \mathcal{N}(0, 1)$:

$$z_i = \mu_i + \sigma_i \epsilon \tag{4}$$

3 β-VAE

β-VAE is a modification of the variational autoencoder (VAE) framework [21, 33] that introduces an adjustable hyperparameter β to the original VAE objective:

$$\mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z}, \beta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta \, D_{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\big) \tag{5}$$

Well chosen values of β (usually $\beta > 1$) result in more disentangled latent representations $\mathbf{z}$. When $\beta = 1$, the β-VAE becomes equivalent to the original VAE framework. It was suggested that the stronger pressure for the posterior $q_\phi(\mathbf{z}|\mathbf{x})$ to match the factorised unit Gaussian prior $p(\mathbf{z})$ introduced by the β-VAE objective puts extra constraints on the implicit capacity of the latent bottleneck $\mathbf{z}$ and extra pressures for it to be factorised while still being sufficient to reconstruct the data $\mathbf{x}$ [14]. Higher values of β necessary to encourage disentangling often lead to a trade-off between the fidelity of β-VAE reconstructions and the disentangled nature of its latent code $\mathbf{z}$ (see Fig. 6 in [14]). This due to the loss of information as it passes through the restricted capacity latent bottleneck $\mathbf{z}$.

4 Understanding disentangling in β-VAE

4.1 Information bottleneck

The β-VAE objective is closely related to the information bottleneck principle [40, 4, 1, 2]:

$$\max[I(Z;Y) - \beta I(X;Z)] \tag{6}$$

where $I(\cdot;\cdot)$ stands for mutual information and β is a Lagrange multiplier. The information bottleneck describes a constrained optimisation objective where the goal is to maximise the mutual information between the latent bottleneck Z and the task Y while discarding all the irrelevant information about Y that might be present in the input X. In the information bottleneck literature, Y would typically stand for a classification task, however the formulation can be related to the auto-encoding objective too [2].

4.2 β-VAE through the information bottleneck perspective

We can gain insight into the pressures shaping the learning of the latent representation $\mathbf{z}$ in β-VAE by considering the posterior distribution $q(\mathbf{z}|\mathbf{x})$ as an information bottleneck for the reconstruction task $\max \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})]$ [2]. The β-VAE training objective (Eq. 5) encourages the latent distribution $q(\mathbf{z}|\mathbf{x})$ to efficiently transmit information about the data points $\mathbf{x}$ by jointly minimising the β-weighted KL term and maximising the data log likelihood.

In β-VAE, the posterior $q(\mathbf{z}|\mathbf{x})$ is encouraged to match the unit Gaussian prior $p(z_i) = \mathcal{N}(0, 1)$. Since the posterior and the prior are factorised (i.e. have diagonal covariance matrix) and posterior samples are obtained using the reparametrization (Eq. 4) of adding scaled independent Gaussian noise $\sigma_i \epsilon_i$ to a deterministic encoder mean μ_i for each latent unit z_i, we can take an information theoretic perspective and think of $q(\mathbf{z}|\mathbf{x})$ as a set of independent additive white Gaussian noise channels z_i, each noisily transmitting information about the data inputs x_n. In this perspective, the KL divergence term $D_{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\big)$ of the β-VAE objective (see Eq. 5) can be seen as an upper bound on the amount of information that can be transmitted through the latent channels per data sample (since it is taken in expectation across the data). The KL divergence is zero when $q(z_i|\mathbf{x}) = p(\mathbf{z})$, i.e μ_i is always zero, and σ_i always 1, meaning the latent channels z_i have zero capacity. The capacity of the latent channels can only be increased by dispersing the posterior means across the data points, or decreasing the posterior variances, which both increase the KL divergence term.

Reconstructing under this bottleneck encourages embedding the data points on a set of representational axes where nearby points on the axes are also close in data space. To see this, following the above, note that the KL can be minimised by reducing the spread of the posterior means, or broadening the posterior variances, i.e. by squeezing the posterior distributions into a shared coding space. Intuitively, we can think about this in terms of the degree of overlap between the posterior distributions across the dataset (Fig. 1). The more they overlap, the broader the posterior distributions will be on average (relative to the coding space), and the smaller the KL divergence can be. However, a greater degree of overlap between posterior distributions will tend to result in a cost in terms of log likelihood due to their reduced average discriminability. A sample drawn from the posterior given one data point may have a higher probability under the posterior of a different data point, an increasingly frequent

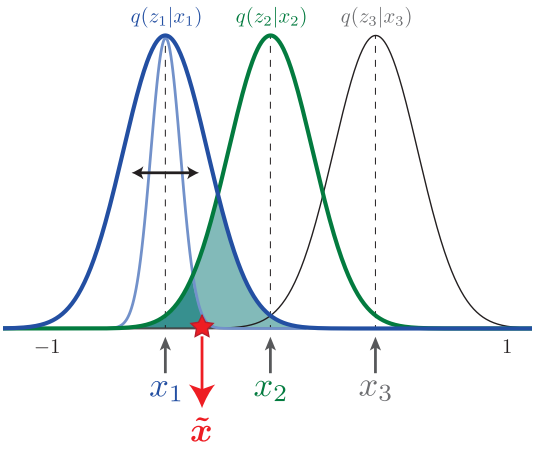


Figure 1: **Connecting posterior overlap with minimizing the KL divergence and reconstruction error.** Broadening the posterior distributions and/or bringing their means closer together will tend to reduce the KL divergence with the prior, which both increase the overlap between them. But, a datapoint $\tilde{x}$ sampled from the distribution $q(z_2|x_2)$ is more likely to be confused with a sample from $q(z_1|x_1)$ as the overlap between them increases. Hence, ensuring neighbouring points in data space are also represented close together in latent space will tend to reduce the log likelihood cost of this confusion.

occurrence as overlap between the distributions is increased. For example, in Figure 1, the sample indicated by the red star might be drawn from the (green) posterior $q(z_2|x_2)$, even though it would occur more frequently under the overlapping (blue) posterior $q(z_1|x_1)$, and so (assuming x_1 and x_2 were equally probable), an optimal decoder would assign a higher log likelihood to x_1 for that sample. Nonetheless, under a constraint of maximising such overlap, the smallest cost in the log likelihood can be achieved by arranging nearby points in data space close together in the latent space. By doing so, when samples from a given posterior $q(z_2|x_2)$ are more likely under another data point such as x_1, the log likelihood $\mathbb{E}_{q(\mathbf{z}_2|\mathbf{x}_2)}[\log p(\mathbf{x}_2|\mathbf{z}_2)]$ cost will be smaller if x_1 is close to x_2 in data space.

4.3 Comparing disentangling in β-VAE and VAE

A representation learned under a weak bottleneck pressure (as in a standard VAE) can exhibit this locality property in an incomplete, fragmented way. To illustrate this, we trained a standard VAE (i.e. with $\beta = 1$) and a β-VAE on a simple dataset with two generative factors of variation: the x and y position of a Gaussian blob (Fig. 2). The standard VAE learns to represent these two factors across four latent dimensions, whereas β-VAE represents them in two. We examine the nature of the learnt latent space by plotting its traversals in Fig. 2, whereby we first infer the posterior $q(\mathbf{z}|\mathbf{x})$, before plotting the reconstructions resulting from modifying the value of each latent unit z_i one at a time in the $[-3, 3]$ range while keeping all the other latents fixed to their inferred values. We can see that the β-VAE represention exhibits the locality property described in Sec. 4.2 since small steps in each of the two learnt directions in the latent space result in small changes in the reconstructions. The VAE represention, however, exhibits fragmentation in this locality property. Across much of the latent space, small traversals produce reconstructions with small, consistent offsets in the position of the sprite, similar to β-VAE. However, there are noticeable representational discontinuities, at which small latent perturbations produce reconstructions with large or inconsistent position offsets. Reconstructions near these boundaries are often of poor quality or have artefacts such as two sprites in the scene.

β-VAE aligns latent dimensions with components that make different contributions to reconstruction We have seen how a strong pressure for overlapping posteriors encourages β-VAE to find a representation space preserving as much as possible the locality of points on the data manifold. However, why would it find representational axes that are aligned with the generative factors of variation in the data? Our key hypothesis is that β-VAE finds latent components which make different contributions to the log-likelihood term of the cost function (Eq. 5). These latent components tend to

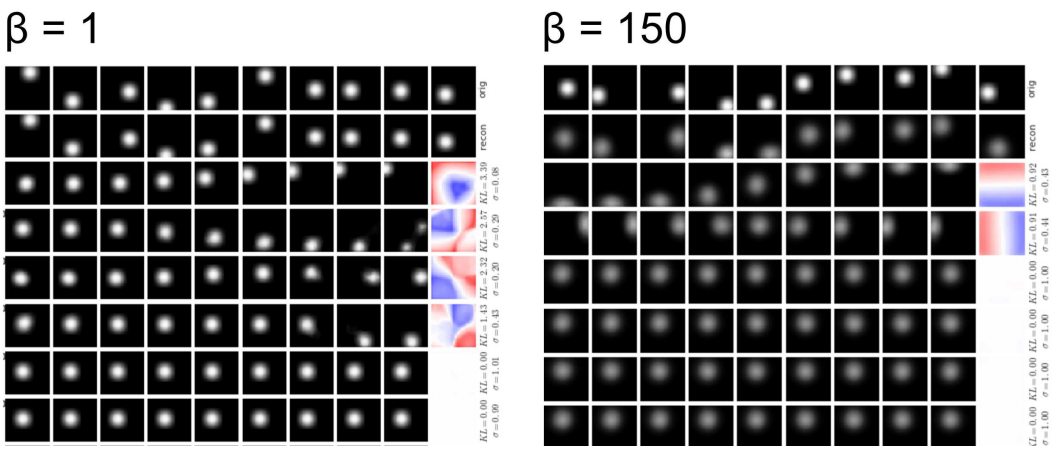


Figure 2: **Entangled versus disentangled representations of positional factors of variation learnt by a standard VAE** ($\beta = 1$) **and** β**-VAE** ($\beta = 150$) **respectively.** The dataset consists of Gaussian blobs presented in various locations on a black canvas. Top row: original images. Second row: the corresponding reconstructions. Remaining rows: latent traversals ordered by their average KL divergence with the prior (high to low). To generate the traversals, we initialise the latent representation by inferring it from a seed image (left data sample), then traverse a single latent dimension (in $[-3, 3]$), whilst holding the remaining latent dimensions fixed, and plot the resulting reconstruction. Heatmaps show the 2D position tuning of each latent unit, corresponding to the inferred mean values for each latent for given each possible 2D location of the blob (with peak blue, -3; white, 0; peak red, 3).

correspond to features in the data that are intuitively qualitatively different, and therefore may align with the generative factors in the data.

For example, consider optimising the β-VAE objective shown in Eq. 5 under an almost complete information bottleneck constraint (i.e. $\beta >> 1$). The optimal thing to do in this scenario is to only encode information about the data points which can yield the most significant improvement in data log-likelihood (i.e. $\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})]$). For example, in the dSprites dataset [27] (consisting of white 2D sprites varying in position, rotation, scale and shape rendered onto a black background), the model might only encode the sprite position under such a constraint. Intuitively, when optimising a pixel-wise decoder log likelihood, information about position will result in the most gains compared to information about any of the other factors of variation in the data, since the likelihood will vanish if reconstructed position is off by just a few pixels. Continuing this intuitive picture, we can imagine that if the capacity of the information bottleneck were gradually increased, the model would continue to utilise those extra bits for an increasingly precise encoding of position, until some point of diminishing returns is reached for position information, where a larger improvement can be obtained by encoding and reconstructing another factor of variation in the dataset, such as sprite scale.

At this point we can ask what pressures could encourage this new factor of variation to be encoded into a distinct latent dimension. We hypothesise that two properties of β-VAE encourage this. Firstly, embedding this new axis of variation of the data into a distinct latent dimension is a natural way to satisfy the data locality pressure described in Sec. 4.2. A smooth representation of the new factor will allow an optimal packing of the posteriors in the new latent dimension, without affecting the other latent dimensions. We note that this pressure alone would not discourage the representational axes from rotating relative to the factors. However, given the differing contributions each factor makes to the reconstruction log-likelihood, the model will try to allocate appropriately differing average capacities to the encoding axes of each factor (e.g. by optimising the posterior variances). But, the diagonal covariance of the posterior distribution restricts the model to doing this in different latent dimensions, giving us the second pressure, encouraging the latent dimensions to align with the factors.

We tested these intuitions by training a simplified model to generate dSprites conditioned on the ground-truth factors, $\mathbf{f}$, with a controllable information bottleneck (Fig. 3). In particular, we wanted

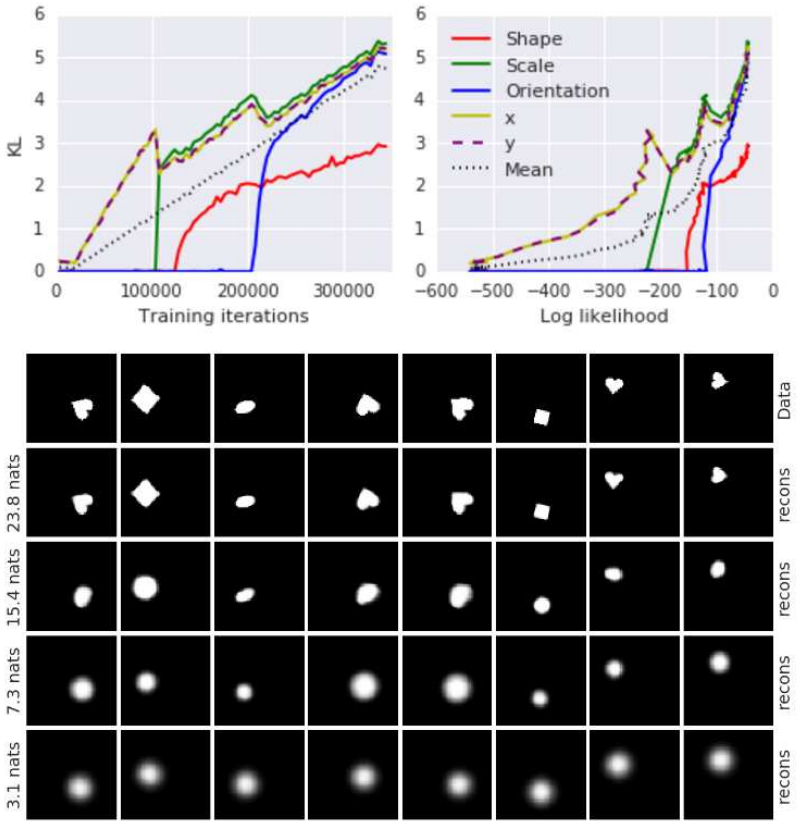


Figure 3: **Utilisation of data generative factors as a function of coding capacity.** Top left: the average KL (in nats) per factor f_i as the training progresses and the total information capacity C of the latent bottleneck $q(\mathbf{z}|\mathbf{f})$ is increased. It can be seen that the early capacity is allocated to positional latents only (x and y), followed by a scale latent, then shape and orientation latents. Top right: same but plotted with respect to the reconstruction accuracy. Bottom: image samples and their reconstructions throughout training as the total information capacity of $\mathbf{z}$ increases and the different latents z_i associated with their respective data generative factors become informative. It can be seen that at 3.1 nats only location of the sprite is reconstructed. At 7.3 nats the scale is also added reconstructed, then shape identity (15.4 nats) and finally rotation (23.8 nats), at which point reconstruction quality is high.

to evaluate how much information the model would choose to retain about each factor in order to best reconstruct the corresponding images given a total capacity constraint. In this model, the factors are each independently scaled by a learnable parameter, and are subject to independently scaled additive noise (also learned), similar to the reparameterised latent distribution in β-VAE. This enables us to form a KL divergence of this factor distribution with a unit Gaussian prior. We trained the model to reconstruct the images with samples from the factor distribution, but with a range of different target encoding capacities by pressuring the KL divergence to be at a controllable value, C. The training objective combined maximising the log likelihood and minimising the absolute deviation from C (with a hyperparameter γ controlling how heavily to penalise the deviation, see Sec. A.2):

$$\mathcal{L}(\theta, \phi; \mathbf{x}(\mathbf{f}), \mathbf{z}, C) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{f})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \gamma \left| D_{KL}\big(q_\phi(\mathbf{z}|\mathbf{f}) \parallel p(\mathbf{z})\big) - C \right| \tag{7}$$

In practice, a single model was trained across of range of C's by linearly increasing it from a low value (0.5 nats) to a high value (25.0 nats) over the course of training (see top left panel in Fig. 3). Consistent with the intuition outlined above, at very low capacities ($C < 5$ nats), the KLs for all the factors except the X and Y position factors are zero, with C always shared equally among X and Y. As expected, the model reconstructions in this range are blurry, only capturing the position of the

original input shapes (see the bottom row of the lower panel in Fig. 3). However, as C is increased, the KLs of other factors start to increase from zero, at distinct points for each factor. For example, starting around $C = 6$ nats, the KL for the scale factor begins to climb from zero, and the model reconstructions become scaled (see 7.3 nats row in lower panel of Fig. 3). This pattern continues until all factors have a non-zero KL and eventually the reconstructions begin to look almost identical to the samples.

5 Improving disentangling in β-VAE with controlled capacity increase

The intuitive picture we have developed of gradually adding more latent encoding capacity, enabling progressively more factors of variation to be represented whilst retaining disentangling in previously learned factors, motivated us to extend β-VAE with this algorithmic principle. We applied the capacity control objective from the ground-truth generator in the previous section (Eq. 7) to β-VAE, allowing control of the encoding capacity (again, via a target KL, C) of the VAE's latent bottleneck, to obtain the modified training objective:

$$\mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z}, C) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \gamma \left| D_{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\big) - C \right| \tag{8}$$

Similar to the generator model, C is gradually increased from zero to a value large enough to produce good quality reconstructions (see Sec. A.2 for more details).

Results from training with controlled capacity increase on coloured dSprites can be seen in Figure 4a, which demonstrate very robust disentangling of all the factors of variation in the dataset and high quality reconstructions.

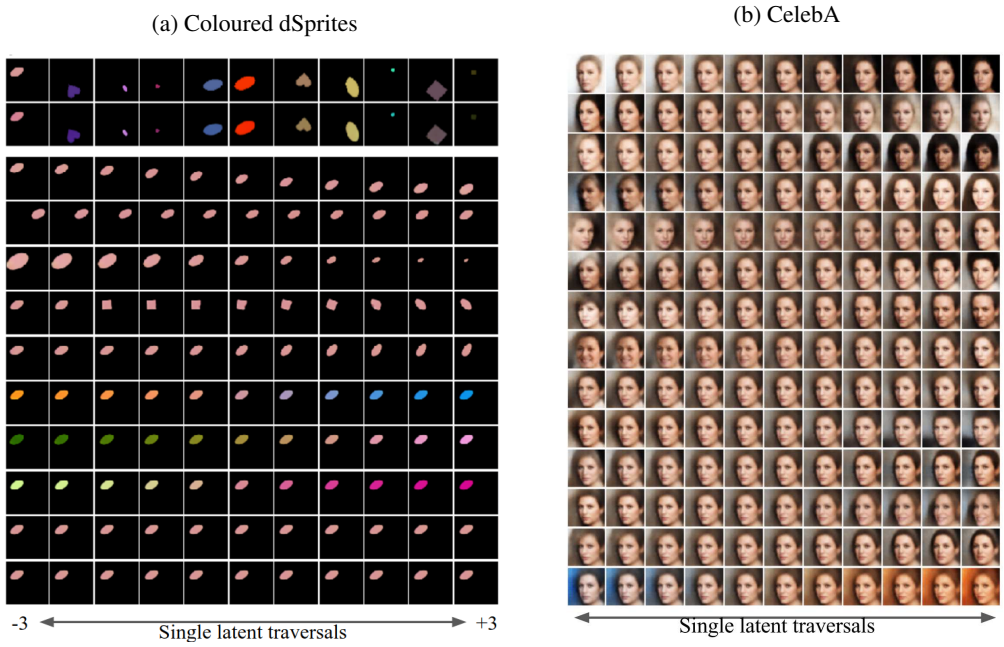


Figure 4: **Disentangling and reconstructions from β-VAE with controlled capacity increase. (a)** Latent traversal plots for a β-VAE trained with controlled capacity increaseon the coloured dSprites dataset. The top two rows show data samples and corresponding reconstructions. Subsequent rows show single latent traversals, ordered by their average KL divergence with the prior (high to low). To generate the traversals, we initialise the latent representation by inferring it from a seed image (left data sample), then traverse a single latent dimension (in $[-3, 3]$), whilst holding the remaining latent dimensions fixed, and plot the resulting reconstruction. The corresponding reconstructions are the rows of this figure. The disentangling is evident: different latent dimensions independently code for position, size, shape, rotation, and colour. **(b)** Latent traversal plots for a β-VAE trained with controlled capacity increaseon the CelebA dataset. Analogous to (a) without with top two rows.

Single traversals of each latent dimension show changes in the output samples isolated to single data generative factors (second row onwards, with the latent dimension traversed ordered by their average KL divergence with the prior, high KL to low). For example, we can see that traversal of the latent with the largest KL produces smooth changes in the Y position of the reconstructed shape without changes in other factors. The picture is similar with traversals of the subsequent latents, with changes isolated to X position, scale, shape, rotation, then a set of three colour axes (the last two latent dimensions have an effectively zero KL, and produce no effect on the outputs).

Furthermore, the quality of the traversal images are high, and by eye, the model reconstructions (second row) are quite difficult to distinguish from the corresponding data samples used to generate them (top row). This contrasts with the results previously obtained with the fixed β-modulated KL objective in [14].

We also trained the same model on the CelebA dataset [26], with latent traversals shown in Figure 4b. In this much richer dataset, it is unclear what the disentangled axes should correspond to. Nonetheless, we can see that traversals of the latent dimensions produce smooth changes in the output samples, with reasonable looking faces in nearly all cases. Furthermore, each traversal appears to generate changes isolated in one or few qualitative features that we might identify intuitively from such face scenes.

6 Conclusion

We have developed new insights into why β-VAE learns an axis-aligned disentangled representation of the generative factors of visual data compared to the standard VAE objective. In particular, we identified pressures which encourage β-VAE to find a set of representational axes which best preserve the locality of the data points, and which are aligned with factors of variation that make distinct contributions to improving the data log likelihood. We have demonstrated that these insight produce an actionable modification to the β-VAE training regime. We proposed controlling the increase of the encoding capacity of the latent posterior during training, by allowing the average KL divergence with the prior to gradually increase from zero, rather than the fixed β-weighted KL term in the original β-VAE objective. We show that this promotes robust learning of disentangled representation combined with better reconstruction fidelity, compared to the results achieved in the original formulation of [14].

References

[1] A. Achille and S. Soatto. Information dropout: Learning optimal representations through noisy computation. *arxiv*, 2016.

[2] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy. Deep variational information bottleneck. *ICLR*, 2016.

[3] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[4] G. Chechik, A. Globerson, N. Tishby, Y. Weiss, and P. Dayan. Information bottleneck for gaussian variables. *Journal of Machine Learning Research*, 6:165–188, 2005.

[5] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv*, 2016.

[6] B. Cheung, J. A. Levezey, A. K. Bansal, and B. A. Olshausen. Discovering hidden factors of variation in deep networks. In *Proceedings of the International Conference on Learning Representations, Workshop Track*, 2015.

[7] T. Cohen and M. Welling. Learning the irreducible representations of commutative lie groups. *arXiv*, 2014.

[8] T. Cohen and M. Welling. Transformation properties of learned visual representations. In *ICLR*, 2015.

[9] G. Desjardins, A. Courville, and Y. Bengio. Disentangling factors of variation via generative entangling. *arXiv*, 2012.

[10] P. Domingos. A few useful things to know about machine learning. *ACM*, 2012.

[11] R. Goroshin, M. Mathieu, and Y. LeCun. Learning to linearize under uncertainty. *NIPS*, 2015.

[12] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. Draw: A recurrent neural network for image generation. *ICML*, 37:1462–1471, 2015.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, pages 770–778, 2016.

[14] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2017.

[15] I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner. DARLA: Improving zero-shot transfer in reinforcement learning. *ICML*, 2017.

[16] I. Higgins, N. Sonnerat, L. Matthey, A. Pal, C. P. Burgess, M. Botvinick, D. Hassabis, and A. Lerchner. Scan: Learning abstract hierarchical compositional visual concepts. *arxiv*, 2017.

[17] G. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders. *International Conference on Artificial Neural Networks*, 2011.

[18] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *ICLR*, 2017.

[19] T. Karaletsos, S. Belongie, and G. Rätsch. Bayesian representation learning with oracle constraints. *ICLR*, 2016.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[21] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *ICLR*, 2014.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *NIPS*, 2012.

[23] T. Kulkarni, W. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. *NIPS*, 2015.

[24] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, pages 1–101, 2016.

[25] Y. LeCun. The next frontier in ai: Unsupervised learning.

[26] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. *ICCV*, 2015.

[27] L. Matthey, I. Higgins, D. Hassabis, and A. Lerchner. dsprites: Disentanglement testing sprites dataset, 2017.

[28] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *ICML*, 2016.

[29] V. Mnih, K. Kavukcuoglu, D. S. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[30] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[31] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. Conditional image generation with pixelcnn decoders. *NIPS*, 2016.

[32] S. Reed, K. Sohn, Y. Zhang, and H. Lee. Learning to disentangle factors of variation with manifold interaction. *ICML*, 2014.

[33] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *ICML*, 32(2):1278–1286, 2014.

[34] K. Ridgeway. A survey of inductive biases for factorial Representation-Learning. *arXiv*, 2016.

[35] O. Rippel and R. P. Adams. High-dimensional probability estimation with deep density models. *arXiv*, 2013.

[36] J. Schmidhuber. Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–869, 1992.

[37] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[38] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CVPR*, 2015.

[39] Y. Tang, R. Salakhutdinov, and G. Hinton. Tensor analyzers. In *Proceedings of the 30th International Conference on Machine Learning, 2013, Atlanta, USA*, 2013.

[40] N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck methods. *arxiv*, 2000.

[41] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *arxiv*, 2017.

[42] W. F. Whitney, M. Chang, T. Kulkarni, and J. B. Tenenbaum. Understanding visual concepts with continuation learning. *arXiv*, 2016.

[43] J. Yang, S. Reed, M.-H. Yang, and H. Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. *NIPS*, 2015.

[44] Z. Zhu, P. Luo, X. Wang, and X. Tang. Multi-view perceptron: a deep model for learning face identity and view representations. In *Advances in Neural Information Processing Systems 27*. 2014.

A Supplementary Materials

A.1 Model Architecture

The neural network models used for experiments in this paper all utilised the same basic architecture. The encoder for the VAEs consisted of 4 convolutional layers, each with 32 channels, 4x4 kernels, and a stride of 2. This was followed by 2 fully connected layers, each of 256 units. The latent distribution consisted of one fully connected layer of 20 units parametrising the mean and log standard deviation of 10 Gaussian random variables (or 32 for the CelebA experiment). The decoder architecture was simply the transpose of the encoder, but with the output parametrising Bernoulli distributions over the pixels. ReLU activations were used throughout. The optimiser used was Adam [20] with a learning rate of 5e-4.

A.2 Training Details

γ used was 1000, which was chosen to be large enough to ensure the actual KL was always close to the target KL, C. For dSprites, C was linearly increased from 0 to 25 nats over the course of 100,000 training iterations, for CelebA it was increased to 50 nats.

Discovering Disentangled Representations with the F Statistic Loss

Karl Ridgeway
Department of Computer Science
University of Colorado, Boulder
karl.ridgeway@colorado.edu

Michael C. Mozer
Department of Computer Science
University of Colorado, Boulder
mozer@colorado.edu

Abstract

We propose and evaluate a novel loss function for discovering deep embeddings that make explicit the categorical and semantic structure of a domain. The loss function is based on the F statistic that describes the separation of two or more distributions. This loss has several key advantages over previous approaches, including: it does not require a margin or arbitrary parameters for determining when distributions are sufficiently well separated, it is expressed as a probability which facilitates its combination with other training objectives, and it seems particularly well suited to disentangling semantic features of a domain, leading to more interpretable and manipulable representations.

In typical classification tasks, the input features—whether images, speech, text, or other measurements—contain only implicit information about category labels, and the job of a classifier is to transform the input features into a representation that makes category labels explicit. The traditional representation has been a *localist* or one-hot encoding of categories, but an alternative approach has recently emerged in which the representation is a *distributed* encoding in a high dimensional space that captures category structure via metric properties of the space. The middle panel of Figure 1 shows a projection of instances from three categories to a two-dimensional space. The projection separates inputs by category and therefore facilitates classification of unlabeled instances via proximity to the category clusters. Such a *deep embedding* also allows new categories to be 'learned' with a few labeled examples that are projected to the embedding space. The literature is somewhat splintered between researchers focusing on deep embeddings which are evaluated via k-shot learning [1, 2, 3] and researchers focusing on k-shot learning who have found deep embeddings to be a useful method [4, 5].

Figure 1 illustrates a fundamental trade off in formulating an embedding. From left to right frames, the intra-class variability increases and the inter-class structure becomes more conspicuous. In the leftmost panel, the clusters are well separated but the classes are all equally far apart. In the rightmost panel, the clusters are highly overlapping and the blue and purple cluster centers are closer to one another than to the yellow. Separating clusters is desirable, but so is capturing inter-class similarity. If this similarity is suppressed, then instances of a novel class will not be mapped in a sensible manner—a manner sensitive to input features, semantic features, and their correspondence. The middle panel reflects a compromise between discarding variability between instances of the same class and preserving relationships among the classes. With this compromise, deep embeddings can

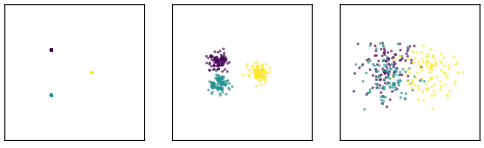


Figure 1: Alternative two-dimensional embeddings of instances of three categories. Points represent instances and color the category label. In the leftmost frame, the points are superimposed on one another.

be used to model hierarchical category structure and can facilitate partitioning the instances along multiple dimensions, e.g., disentangling content and style [6].

The trade off in Figure 1 points to a challenge in constructing deep embeddings. Some existing methods aim to perfectly separate categories in the training set [1], which may not be appropriate if there are labeling errors or noise in the data. Other methods require a margin or other parameter to determine how well separated the categories should be in order to prevent overfitting [7, 2, 3, 8]. We propose a new method that automatically balances the trade off using the currency of probability and statistical hypothesis testing. It also manages to align dimensions of the embedding space with categorical and semantic features, thereby facilitating the disentangling of representations.

1 Using the F statistic to separate classes

For expository purposes, consider two classes, $\boldsymbol{C} = \{1, 2\}$, having n_1 and n_2 instances, which are mapped to a one-dimensional embedding. The embedding coordinate of instance j of class i is denoted z_{ij}. The goal of any deep embedding procedure is to separate the coordinates of the two classes. In our approach, we will quantify the separation via the probability that the true class means in the underlying environment, μ_1 and μ_2, are different from one another. Our training goal can thus be formulated as minimizing $\Pr(\mu_1 = \mu_2 \mid \boldsymbol{s}(\boldsymbol{z}), n_1, n_2)$, where $\boldsymbol{s}(\boldsymbol{z})$ denotes summary statistics of the labeled embedding points. This posterior is intractable, so instead we operate on the likelihood $\Pr(\boldsymbol{s}(\boldsymbol{z}) \mid \mu_1 = \mu_2, n_1, n_2)$ as a proxy.

We borrow a particular statistic from analysis of variance (ANOVA) hypothesis testing for equality of means. The statistic is a ratio of between-class variability to within-class variability:

$$s = \tilde{n} \frac{\sum_i n_i (\bar{z}_i - \bar{\bar{z}})^2}{\sum_{i,j} (z_{ij} - \bar{z}_i)^2}$$

where $\bar{z}_i = \langle z_{ij} \rangle$ and $\bar{\bar{z}} = \langle \bar{z}_i \rangle$ are expectations and $\tilde{n} = n_1 + n_2 - 2$. Under the null hypothesis $\mu_1 = \mu_2$ and an additional normality assumption, $z_{ij} \sim \mathcal{N}(\mu, \sigma^2)$, our statistic s is a draw from a Fisher-Snedecor (or F) distribution with degrees of freedom 1 and $\tilde{n}$, $S \sim F_{1,\tilde{n}}$. Large s indicate that embeddings from the two different classes are well separated relative to two embeddings from the same class, which is unlikely under $F_{1,\tilde{n}}$. Thus, the CDF of the F distribution offers a measure of the separation between classes:

$$\Pr(S < s | \, \mu_1 = \mu_2, \tilde{n}) = I\left(\frac{s}{s + \tilde{n}}, \frac{1}{2}, \frac{\tilde{n}}{2}\right) \tag{1}$$

where I is the regularized incomplete beta function, which is differentiable and thus can be incorporated into an objective function for gradient-based training.

Several comments on this approach. First, although it assumes the two classes have equal variance, the likelihood in Equation 1 is fairly robust against inequality of the variances as long as $n_1 \approx n_2$. Second, the F statistic can be computed for an arbitrary number of classes; the generalization of the likelihood in Equation 1 is conditioned on *all* class instances being drawn from the same distribution. Because this likelihood is a very weak indicator of class separation, we restrict our use of the F statistic to class pairs. Third, this approach is based entirely on *statistics* of the training set, whereas every other deep-embedding method of which we are aware uses training criteria that are based on individual instances. For example, the triplet loss [7] attempts to ensure that for specific triplets $\{z_{11}, z_{12}, z_{21}\}$, z_{11} is closer to z_{12} than to z_{21}. Objectives based on specific instances will be more susceptible to noise in the data set and may be more prone to overfitting.

1.1 From one to many dimensions

Our example in the previous section assumed one-dimensional embeddings. We have explored two extensions of the approach to many-dimensional embeddings. First, if we assume that the Euclidean distances between embedded points are gamma distributed—which turns out to be a good empirical approximation at any stage of training—then we can represent the numerator and denominator in the F statistic as sums of gamma random variables, and a variant of the unidimensional separation measure (Equation 1) can be used to assess separation based on Euclidean distances. Second, we can apply the unidimensional separation measure for multiple dimensions of the many-dimensional embedding space. We focus on the latter approach in this article.

For a given class pair (α, β), we can compute $\Pr(S < s | \mu_{1k} = \mu_{2k})$ for each dimension k of the embedding space We select a set, $\boldsymbol{D}_{\alpha,\beta}$, of the d dimensions with largest $\Pr(S < s | \mu_{1k} = \mu_{2k})$ i.e., the dimensions that are best separated already. Although it is important to separate classes, they needn't be separated on *all* dimensions because the pair may have semantic similarity or equivalence along some dimensions. The pair is separated if they can be distinguished reliably on a subset of dimensions.

For a training set or a mini-batch with multiple instances of a set of classes $\boldsymbol{C}$, our embedding objective is to maximize the joint probability of separation for all class pairs (α, β) on all relevant dimensions, $\boldsymbol{D}_{\alpha,\beta}$. Framed as a loss, we minimize the log probability:

$$\mathcal{L}_F = -\sum_{\{\alpha,\beta\}\in\boldsymbol{C}} \sum_{k\in\boldsymbol{D}_{\alpha,\beta}} \ln \Pr(S < s | \mu_{1k} = \mu_{2k})$$

This *F-statistic loss* has four desirable properties. First, the gradient rapidly drops to zero once classes become reliably separated on at least d dimensions, leading to a natural stopping criterion; the degree of separation obtained is related to the number of samples per class. Second, in contrast to other losses, the F-statistic loss is not invariant to rotations in the embedding space; this focus on separating along specific dimensions tends to yield disentangled features when the class structure is factorial or compositional. Third, embeddings obtained are relatively insensitive to the one free parameter, d. Fourth, because the loss is expressed in the currency of probability it can readily be combined with additional losses expressed similarly (e.g., a reconstruction loss framed as a likelihood). The following sections demonstrate the advantages of the F-statistic loss for identity classification and for disentangling attributes related to identity.

2 Identity Classification

In this section, we demonstrate the advantages of the F-statistic loss over state-of-the-art methods on identity classification. The task involves matching a person from a wide-angle, full-body photograph, taken at various angles and poses. We evaluate two data sets—CUHK03 [9] and Market-1501 [10]—following the methodology of [1]. Five-fold cross validation is performed for CUHK03, and a single train/test split used for Market-1501.

Training Details. Following [1], we use the Deep Metric Learning [3] architecture with a 500-dimensional embedding. All nets were trained using the ADAM [11] optimizer, with a learning rate of 10^{-4}. A validation set was withheld from the training set, and used for early stopping. To construct a mini-batch for training, we randomly select 12 identities, with up to 10 samples of each identity, as in [1]. In addition to the F-statistic loss, we evaluated histogram [1], triplet [2], and binomial deviance [12] losses. For the triplet loss, we use all triplets in the minibatch. For the histogram loss and binomial deviance losses, we use all pairs. For the F-loss, we use all class pairs. The triplet loss is trained and evaluated using L_2 distances. The F-statistic loss is evaluated using L_2 distances. As in [1], embeddings obtained discovered by the histogram and binomial-deviance losses are constrained to lie on the unit hypersphere; cosine distance is used for training and evaluation. For the F-statistic loss, we determined the best value of d, the number of dimensions to separate, using the validation set of the first split. Performance is relatively insensitive to d for $2 < d < 100$.

Results. Figure 2 reports Recall@k accuracy, the performance metric used in earlier work. For each query image in the test set, we compute the distance of its embedding vector to the embedding vectors in the remainder of the test set. A query returns a 1 if one of the k nearest neighbors in the embedding space is of the same class as the query image, 0 otherwise. Recall@k is the percentage

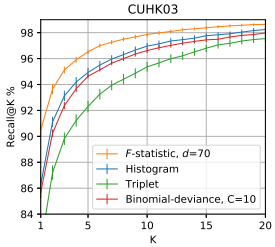


Figure 2: Recall@k for the CUHK03 dataset and four deep-embedding losses. For CUHK03, each line indicates the mean Recall@k over cross-validation splits, and vertical bars indicate ± 1 standard error of the mean.

of queries that return a 1. The F-statistic loss leads to reliably better accuracy, especially for low k. On CUHK03, the F-statistic loss obtains a Recall@1 accuracy of $90.5\% \pm 0.4\%$, compared to next best, histogram loss, $86.3\% \pm 0.6\%$. On Market-1501, the single train-test split yields comparable performance for all losses for small k, e.g., Recall@1 accuracy is 66.51% for the histogram loss, 65.87% for the triplet loss, 65.75% for the F-statistic loss, and 65.45% for binomial deviance loss.

3 Disentangling Identity Attributes

Next, we show that the F-statistic loss obtains disentangled embeddings—embeddings whose dimensions are aligned with the categorical and semantic features of the input data. We explore disentangling with a data set of video game sprites—60×60 pixel color images of game characters viewed from various angles and in a variety of poses [13]. The identity of the game characters is composed of 7 attributes—body, arms, hair, gender, armor, greaves, and weapon—each with 2–5 distinct values, leading to 672 total unique identities which can be instantiated in various viewing angles and poses.

We used the encoder architecture of Reed et al. [13] as well as their embedding dimensionality of 22. We evaluated with five-fold cross validation, splitting by identity and including all variations in viewing angle and pose. A portion of the training set was reserved to determine when to stop training based on Recall@1 performance. For these experiments, we compare the F-statistic loss to the triplet loss; other losses using L_p norm distances should yield similar results.

The sprite dataset is factorial: every combination of attribute-values is present. An ideal disentangled representation will also be factorial, wherein all pairs of dimensions are statistically independent. However, due to the fact that the embedding dimensionality may allow for redundancy, simply measuring mutual information will not reveal disentangled structure: if one embedding is more compact than another, it will allow for more redundancy and consequently higher mutual information. As an alternative to mutual information, we measure how well each embedding dimension predicts each identity-attribute value (e.g., hair=blond, weapon=spear); in a disentangled representation, single dimensions should be highly predictive of these values. For each value, we assess how well each embedding dimension discriminates the given value from other values of the attribute, and record the AUC of the most predictive embedding dimension. There are in total 17 (nonredundant) attribute values, and five cross-validation splits, so we record 85 AUCs for each training loss. AUC is based on the entire dataset to ensure adequate coverage over all attribute values. Figure 3 shows the distribution of AUCs for embeddings based on the triplet, histogram, and F-statistic losses. The embeddings trained using the F-statistic loss are more likely to include dimensions that are aligned with the generative attributes of the domain (i.e., AUC close to 1). This property is robust for moderate values of d.

4 Discussion and Future Work

The F-statistic loss is a novel approach to learning deep embeddings that uses only summary statistics to judge embedding quality, in contrast to approaches that examine the relationships among the individual embedding points. Our approach beats state-of-the-art performance on the "person re-identification" task. Our approach also yields better disentangling of factors that compose identity, leading to more interpretable representations.

We are presently investigating the use of this loss for disentangling content and style (or, identity and non-identity) by incorporating an additional reconstruction loss to ensure that the combined content+style representation preserves information in the input [14, 15]. We further expect to improve

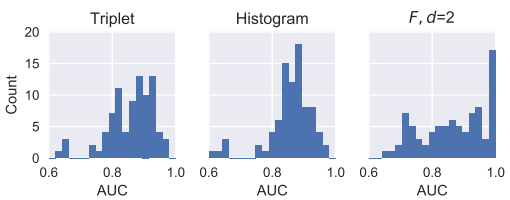


Figure 3: Predicting semantic attributes from single embedding dimensions. AUC distributions for triplet, histogram, and $d = 2$ for the F-statistic loss.

the disentangling of content and style by inverting the F-statistic loss for the style component of the embedding to reduce class separation. Finally, we are evaluating the content-style decompositions obtained with the F-statistic loss to those obtained by other losses, in an effort to demonstrate that the F-statistic decompositions are superior for image synthesis and for generating augmented data sets.

References

[1] Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. *Advances in Neural Information Processing Systems*, pages 4170–4178, 2016.

[2] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[3] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Deep metric learning for person re-identification. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 34–39. IEEE, 2014.

[4] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In U. V. Luxburg, I. Guyon, S. Bengio, H. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages xxx–xxx. Curran Associates, Inc., 2017.

[5] E. Triantafillou, R. Zemel, and R. Urtasan. Few-shot learning through an information retrieval lens. In U. V. Luxburg, I. Guyon, S. Bengio, H. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages xxx–xxx. Curran Associates, Inc., 2017.

[6] J B Tenenbaum and W T Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000.

[7] S Chopra, R Hadsell, and LeCun Y. Learning a similiarty metric discriminatively, with application to face verification. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 349–356, 2005.

[8] Hyun Oh Song, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding query retrieval. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4004–4012, 2016.

[9] Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 152–159, 2014.

[10] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1116–1124, 2015.

[11] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[12] Dong Yi, Zhen Lei, and Stan Z. Li. Deep metric learning for practical person re-identification. *ICPR*, 11(4):1–11, 2014.

[13] Scott E. Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. *Advances in Neural Information Processing Systems*, pages 1252–1260, 2015.

[14] Salah Rifai, Yoshua Bengio, Aaron Courville, Pascal Vincent, and Mehdi Mirza. Disentangling factors of variation for facial expression recognition. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7577 LNCS(PART 6):808–822, 2012.

[15] Tejas D Kulkarni, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547, 2015.

Disentangling Video with Independent Prediction

William F. Whitney and Rob Fergus
Department of Computer Science
New York University
{wwhitney,fergus}@cs.nyu.edu

Abstract

We propose an unsupervised variational model for disentangling video into independent factors, i.e. each factor's future can be predicted from its past without considering the others. We show that our approach often learns factors which are interpretable as objects in a scene.

1 Introduction

Deep neural networks have delivered impressive performance on a range of perceptual tasks, but their distributed representation is difficult to interpret and poses a challenge for problems involving reasoning. Motivated by this, the deep learning community has recently explored methods [1–4,6–8,12,15–17,20,21] for learning distributed representations which are *disentangled*, i.e. a unit (or small group) within the latent feature vector are exclusively responsible for capturing distinct concepts within the input signal. This work proposes such an approach for the video domain, where the temporal structure of the signal is leveraged to automatically separate the input into factors that vary independently of one another over time. Our results demonstrate that these factors correspond to distinct objects within the video, thus providing a natural representation for making predictions about future motions of the objects and subsequent high-level reasoning tasks.

Related work. [22] leverage structure at different time-scales to factor signals into independent components. Our approach can handle multiple factors at the same time-scale, instead relying on prediction as the factoring mechanism. Outside of the video domain, [1] proposed to disentangle factors that tend to change independently and sparsely in real-world inputs, while preserving information about them. [15] learned to disentangle factors of variation in synthetic images using weak supervision, and [21] extended the method to be fully unsupervised. Similar to our work, both [3] and [4] propose unsupervised schemes for disentangling video, the latter using a variational approach. However, ours differs in that it uncovers general factors, rather than specific ones like identity and pose or static/dynamic features as these approaches do.

2 Generative model and inference

The intuition behind our model is simple: in any real-world scene, most objects do not physically interact with one another, so their motion can be modeled independently. To find a representation of videos with these same independences, we introduce an approach based on a temporal version [14] of the variational auto-encoder [10], shown in Figure 1. In this model, each video X comprises a sequence of frames $x_1 ... x_n$. $Z_t = z_t^1 ... z_t^k$ represents all the latent factors at time t, where each factor z_t^i is represented by a vector. Each of these factors evolve independently from one another and combine to produce the observation x_t for timestep t. Instead of directly maximizing the likelihood $p(X;\theta)$, we use variational inference [5] to optimize the evidence lower bound (ELBO), that is $\theta^*, \phi^* = \arg\max_{\theta,\phi} \mathcal{L}(\theta, \phi; X)$:

$$\mathop{\mathbb{E}}_{Z\sim q_\phi(Z|X)} \log p_\theta(X|Z) - D_{KL}(q_\phi(Z_1|X)||p_\theta(Z_1)) - \sum_{t=2}^{n} \mathop{\mathbb{E}}_{\substack{Z_{t-1}\sim \\ q(Z_{t-1}|Z_{t-2},X)}} D_{KL}(q_\phi(Z_t|Z_{t-1},X)||p_\theta(Z_t|Z_{t-1}))$$

Learning Disentangled Representations Workshop. NIPS 2017. Long Beach, CA, USA.

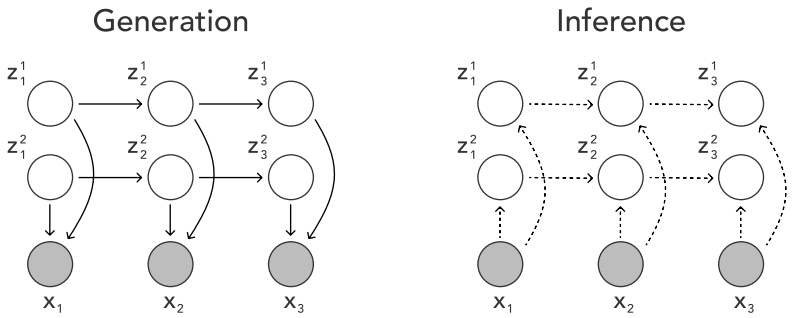


Figure 1: **Left:** Our generative model. Several latent variables combine to produce each observation, and each variable evolves through time independently of the others, just like objects which do not collide. Each observation x_t is given by a decoder which produces $p_\theta(x_t|z_t^1...z_t^k)$. **Right:** Our variational inference procedure gives $q_\phi(z_t^1...z_t^k|x_t, z_{t-1}^1...z_{t-1}^k) \approx p(z_t^1...z_t^k|x_t, z_{t-1}^1...z_{t-1}^k)$.

distribution	parameterization	parameter sharing?
$p_\theta(z_1^i)$	$\mathcal{N}(0, \mathbb{I})$	N/A (but same for each i)
$p_\theta(z_t^i\|z_{t-1}^i)$	MLP$(d \to 128, 128 \to 128, 128 \to d)$	across t (different params for each i)
$p_\theta(x_t\|Z_t)$	DCGAN generator	across t
$q_\phi(Z_t\|Z_{t-1}, x_t)$	DCGAN discriminator	across t

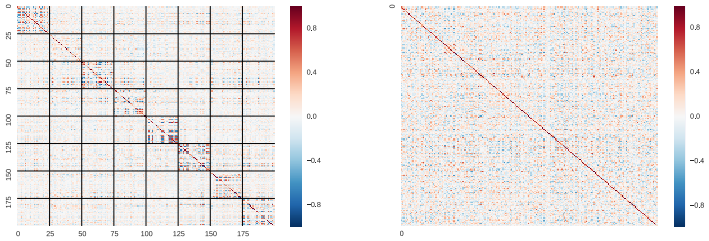
Table 1: The parameterization for each of the distributions in our model. Each latent factor has its own transition function, but all modules are shared across all timesteps. The variable d is the dimension of the entire latent space including all factors. DCGAN refers to the architecture used in [14]. For more details, please see Appendix A.

For a derivation, see Appendix B. This lower bound naturally splits into two factors. The first, $p_\theta(X|Z)$, is the log-likelihood of the data X under our model when sampling from the approximate posterior, that is the "reconstruction" of X from Z. The second factor is the KL divergence between the learned prior $p_\theta(Z)$ and the approximate posterior $q_\phi(Z|X)$. It represents how far the predictions given by p are from the inferred values given by q; it is the prediction error in the latent space. Our goal is to optimize the space of Z to make both reconstruction and prediction possible.

For the variational approximation, we choose $q_\phi(Z|X) = \prod_{t=1}^n q(Z_t|Z_{t-1}, x_t; \phi)$ which, analogous to a Kalman filter, considers only the past state and current observation. This approximation marginalizes over the future, in that $q(Z_t|Z_{t-1}, x_t)$ must encode sufficient information to allow correct predictions of the future as well as fit the prior and the current observation. This allows us to do inference on a single frame and ensure our representation retains as much information about the future as possible. If we wish to use our learned representation for a downstream task, we may discard the generative model and use the inference network alone. This inference network can provide a factorized representation given a single image or use a sequence of images to produce increasingly tight estimates of the latent variables.

Neural network parameterization. We choose all of the distributions in our model to be Gaussian with diagonal covariance. To allow our model to scale to complex nonlinear observations and dynamics, we parameterize each distribution with a neural network. Table 1 describes each of these parameterizations. As each distribution is diagonal Gaussian, each network produces two outputs, corresponding to the mean and the variance of each distribution.

Training. This model can be thought of as a series of variational autoencoders with a learned prior, and the training procedure is largely similar to [10]. At each timestep t in a sequence, we compute the prior $p_\theta(Z_t|Z_{t-1})$ over the latent space, using $\mathcal{N}(0, \mathbb{I})$ for $t = 0$. We infer the approximate posterior $q_\phi(Z_t|Z_{t-1}, x_t)$ by observing x_t and compute the KL divergence $D_{KL}(q||p)$. We then sample $Z_t \sim q_\phi(Z_t|Z_{t-1}, x_t)$ using the reparameterization trick and compute $\log p_\theta(x_t|Z_t)$. At the end of a sequence we update our parameters θ, ϕ by backprop to maximize the ELBO (defined above).



(a) Factored model (b) Entangled model

Figure 2: Heatmap of correlation between latent variables (shown as blocks) at time t (y-axis) vs. at time $t+1$ (x-axis) on the 5th Avenue dataset. The block-diagonal structure exhibited here reflects that z_{t+1}^i largely depends on its own previous state z_t^i rather than on the other latent factors. The entangled model does not appear to have structure in its latent space. These plots were generated using models with two-layer transition networks; with deeper transitions there is almost zero linear dependence between any pair of latent units.

3 Experiments

Datasets. We apply this model to two video datasets: the widely-used moving MNIST [19] and a new dataset of real-world videos of 5th Avenue recorded from above, which was collected by the authors. More details about these datasets, including a sample frame from 5th Avenue, can be found in Appendix C.

Baseline. In each of these evaluations we compare with a model which is identical to the factored model except that its transition function $p(Z_t|Z_{t-1})$ is not factored (referred to as the *entangled* model in our experiments). It can be viewed as a special case of our factored model which has a single high-dimensional latent factor, and its latent space has the same total dimension as the corresponding factored model in each experiment. This comparison is intended to be as tight as possible, with any differences between the factored model and the baseline coming exclusively from the factorization in the latent space.

3.1 Evaluation

Lower bound. We compare the variational lower bound achieved by our factorized model with that of a non-factorized but otherwise identical model. These experiments reveal the price in terms of data fidelity that we pay for representing the data as independently-changing factors. Table 2 shows that the factored model achieves a lower bound on par with the entangled model.

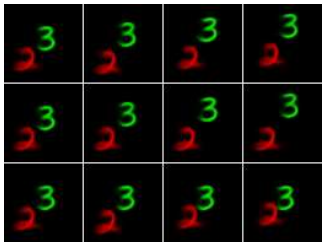
Correlation structure. By plotting the correlation between samples from the approximate posterior over the latent variables at time t and at time $t+1$, we may observe whether our model has been able to learn a representation for which z_{t+1}^i really does only depend on z_t^i and not on the other latent factors. Note that this only captures the *linear* dependence of these variables; this analysis helps to illustrate the structure of our latent space, but should not be considered definitive. Figure 2 shows that each latent factor is much more correlated with its own previous state than with other latent factors.

Approximate mutual information. We use Kraskov's method for estimating mutual information [13] to approximate $\mathbb{I}(z_{t+1}^i; z_t^i)$ and compare it to $\mathbb{I}(z_{t+1}^i; z_t^{j\neq i})$. A latent factor z_t^i should be much more informative about its own future z_{t+1}^i than a different factor $z_t^{j\neq i}$ is. The results, shown in Table 2, reveal that the evolution of each factor in the factored models depends almost exclusively on their past.

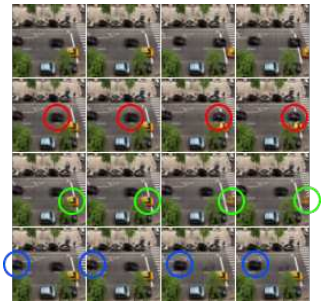
For the entangled models, which do not have separate factors, there is no *a priori* subdivision into high- and low-mutual-information segments of units. The reported scores were generated by creating 20 random partitionings of the latent units into k factors, then reporting the mutual information numbers for the partitioning that had the greatest difference between same-factor and cross-factor information.

dataset, model	$\tilde{\mathbb{I}}(z^i_{t+1}; z^i_t)$	$\tilde{\mathbb{I}}(z^i_{t+1}; z^{j\neq i}_t)$	ELBO
moving MNIST, two factors	4.61	0.75	-2902
moving MNIST, entangled	4.66	2.61	-2896
5th Ave, eight factors	2.26	0.28	6830
5th Ave, entangled	2.08	0.38	6800

Table 2: Mutual information estimates and ELBO. In the factored model, the previous value of the same latent variable is substantially more predictive of its current value than are the previous values of any other variables. This shows that our model is actually factorizing. See Sec. 3.1 for details. The ELBO $\mathcal{L}(\theta, \phi; X)$ shows that our model pays a small or nonexistent price for factoring the latent space into independently-evolving components.



(a) Factorized reconstruction of moving MNIST



(b) Factorized reconstruction of 5th Avenue

Figure 3: **Left, top row:** reconstruction of the input sequence using all factors. **Left, middle row:** reconstructing the sequence varying only the first latent factor (of two). Only the green three moves, showing that factor 1 exactly represents the green digit. **Left, bottom row:** reconstructing the sequence varying only the second latent factor. Evolving factor two only affects the pose of the red digit. **Right, top row:** reconstruction of the input sequence using all latents. **Right, other rows:** reconstructions only varying a single latent factor. Several factors correspond to moving a single car in the image. Note that in the last row, the model removes the top-right black car; since the top-left car is moving forward quickly, it predicts that there must not be an obstacle in its way.

The entangled models show much more cross-factor information than the factored models in our tests on moving MNIST, where we use two latent factors. However, as the number of factors increases, the average information between a pair of factors naturally diminishes. As a result on 5th Avenue, where we subdivide the latent units into eight factors, the entangled model shows cross-information almost as low as the factored model.

Independent generations. Finally, we evaluate qualitatively the representations learned by our model. We infer the approximate posterior $q(Z|X)$, then set all of the latent variables fixed at their $t = 2$ values given by $q(Z_2|x_{1,2})$. We then produce a sequence of generations by picking a single variable z^i to vary, then for each timestep $t = 2...n$ drawing a sample from $p(x|z^1_2...z^{i-1}_2, \boldsymbol{z^i_t}, z^{i+1}_2...z^k_2)$ (note the single bolded factor $\boldsymbol{z^i_t}$ varying with t). That is, we hold all but one of the latent variables fixed and allow the single one to vary with the posterior. This allows us to see exactly what that single variable represents in this video. The images generated by this process are shown in Figure 3.

4 Discussion

By taking advantage of the structure present in video, our model can pull apart latent factors which change independently and produce a representation composed of semantically meaningful variables. The approach is conceptually simple and based on the insight that if two objects do not interact, they can be predicted independently. In the future we hope to apply a richer family of approximations to scale to more complex data.

References

[1] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35, 8 (2013), 1798–1828.

[2] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. 2016. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341* (2016).

[3] Emily Denton and Vighnesh Birodkar. 2017. Unsupervised learning of disentangled representations from video. *CoRR* abs/1705.10915, (2017).

[4] Will Grathwohl and Aaron Wilson. 2016. Disentangling space and time in video with hierarchical variational auto-encoders. *arXiv preprint arXiv:1612.04440* (2016).

[5] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. 2013. Stochastic variational inference. *The Journal of Machine Learning Research* 14, 1 (2013), 1303–1347.

[6] Wei-Ning Hsu, Yu Zhang, and James R. Glass. 2017. Unsupervised learning of disentangled and interpretable representations from sequential data. *CoRR* abs/1709.07902, (2017).

[7] Aapo Hyvarinen and Hiroshi Morioka. 2016. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. In *NIPS*.

[8] Wu Janner M. 2017. Learning to generalize intrinsic images with a structured disentangling autoencoder. In *NIPS*.

[9] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[10] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[11] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-normalizing neural networks. *arXiv preprint arXiv:1706.02515* (2017).

[12] Mathias Berglund Klaus Greff Antti Rasmus and Juergen Schmidhuber. 2016. Deep unsupervised perceptual grouping. In *NIPS*.

[13] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Physical review E* 69, 6 (2004), 066138.

[14] Rahul G Krishnan, Uri Shalit, and David Sontag. 2017. Structured inference networks for nonlinear state space models. In *AAAI*, 2101–2109.

[15] Tejas D Kulkarni, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum. 2015. Deep convolutional inverse graphics network. In *Advances in neural information processing systems*, 2530–2538.

[16] Ulrich Paquet Marco Fraccaro Simon Kamronn. 2017. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *NIPS*.

[17] Jan-Willem Van de Meent N. Siddharth Brooks Paige. 2017. Learning disentangled representations with semi-supervised deep generative models. In *NIPS*.

[18] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[19] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. 2015. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, 843–852.

[20] Valentin Thomas, Jules Pondard, Emmanuel Bengio, Marc Sarfati, Philippe Beaudoin, Marie-Jean Meurs, Joelle Pineau, Doina Precup, and Yoshua Bengio. 2017. Independently controllable factors. *arXiv 1708.01289* (2017).

[21] William F Whitney, Michael Chang, Tejas Kulkarni, and Joshua B Tenenbaum. 2016. Understanding visual concepts with continuation learning. *arXiv preprint arXiv:1602.06822* (2016).

[22] Laurenz Wiskott and Terrence J Sejnowski. 2002. Slow feature analysis: Unsupervised learning of invariances. *Neural computation* 14, 4 (2002), 715–770.

Appendix A: Network architecture details

All models are trained using the ADAM optimizer [9] with a learning rate of $3e{-}4$.

Inference network

For the inference network, which parameterizes the function $q(Z_t|Z_{t-1}, x_t)$, we use an architecture derived from the DCGAN discriminator [18]. We use the discriminator architecture to encode the input image x, then add an additional input to take in the value of Z_{t-1}. We pass the inference network the *predicted values* $p_\theta(Z_t|Z_{t-1})$ instead of having it do inference directly from Z_{t-1}. We found that this greatly sped up training as the inference network doesn't have to learn the transition function in order to fit to it. This is equivalent to sharing parameters between the transition network and the inference network, though the transition parameters are not updated here.

DCGAN image encoder:

```
Conv2d(3, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True)
Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
Conv2d(512, 32, kernel_size=(4, 4), stride=(1, 1))
```

Combining information about Z_{t-1} with information about x_t:

```
transformed_x = Linear(32 -> z_dim)(encoder_output)
transformed_mu = Linear(z_dim -> z_dim)( mu(Z_{t} | Z_{t-1}) )
transformed_sigma = Linear(z_dim -> z_dim)( sigma(Z_{t} | Z_{t-1}) )
latent = Linear(z_dim * 3 -> z_dim)(transformed_x, transformed_mu, transformed_sigma)
output_mu = Linear(z_dim -> z_dim)(latent)
output_sigma = Linear(z_dim -> z_dim)(latent)
```

where `mu(Z_{t} | Z_{t-1})` and `sigma(Z_{t} | Z_{t-1})` are the mean and variance vectors respectively of the prediction $p_\theta(Z_t|Z_{t-1})$ and `z_dim` is the number of latent factors times the dimensionality of each factor. `output_mu` and `output_sigma` are the mean and diagonal covariance of the approximate posterior, and `output_sigma` is actually output as $\log \sigma^2$ for numerical reasons. Each layer is followed by a Leaky ReLU activation.

At $t = 0$, when there is no Z_{t-1}, we pass all-zero vectors instead.

Generator network

The generator network takes in a latent vector Z and produces a pixelwise mean for an output image. It has this form:

```
Linear (z_dim -> z_dim)
ConvTranspose2d(z_dim, 512, kernel_size=(4, 4), stride=(1, 1))
BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
ConvTranspose2d(128, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True)
ConvTranspose2d(64, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
```

where `z_dim` is the number of latent factors times the dimensionality of each factor. Each layer is followed by a ReLU activation. We use a fixed variance in our Normal observation model of 0.25 for moving MNIST and 0.05 for 5th Avenue. This variance hyperparameter may be tuned to balance the tradeoff between fitting the predictions and making tight reconstructions.

Transition network

Each latent factor has its own transition function $p_\theta(z_t^i|z_{t-1}^i)$. Each of these has the following form:

```
Linear (latent_dim -> 128)
Linear (128 -> 128)
Linear (128 -> 128)
Linear (128 -> 128)
Linear (128 -> latent_dim * 2)
```

where `latent_dim` is the dimensionality of a single latent factor and each layer is followed by a SELU activation [11]. The `latent_dim * 2` output is for the mean and (diagonal) variance vectors for the Normal distribution. The variance is produced as $\log \sigma^2$ for numerical reasons.

Appendix B: Deriving the ELBO

For this section the factored form of the transitions $p_\theta(Z_t|Z_{t-1}) = \prod_{i=1}^k p_\theta(z_t^i|z_{t-1}^i)$ is not relevant. As such our development here will use the more general non-factored form, and we can substitute in our factored special case later. For simplicity of notation we will use $p(\cdot)$ in place of $p_\theta(\cdot)$. Likewise we use $q(\cdot)$ in place of $q_\phi(\cdot)$ to represent our variational approximation function with parameters ϕ.

We begin with the form of our latent-variable generative model.

$$\begin{aligned}\log p(X) &= \log \int p(X|Z)p(Z)dZ \\ &= \log \int p(X|Z)p(Z_1)p(Z_2|Z_1)...p(Z_{t_{max}}|Z_{t_{max}-1}...Z_1)dZ\end{aligned}$$

Since the series Z is Markov,

$$= \log \int p(X|Z)p(Z_1)p(Z_2|Z_1)...p(Z_{t_{max}}|Z_{t_{max}-1})dZ$$

We introduce our variational auxiliary functions:

$$\begin{aligned}&= \log \int p(X|Z)p(Z_1)p(Z_2|Z_1)...p(Z_{t_{max}}|Z_{t_{max}-1})\frac{q(Z_1|X)q(Z_2|Z_1,X)...q(Z_{t_{max}}|Z_{t_{max}-1},X)}{q(Z_1|X)q(Z_2|Z_1,X)...q(Z_{t_{max}}|Z_{t_{max}-1},X}))dZ \\ &= \log \int p(X|Z)\frac{p(Z_1)}{q(Z_1|X)}\frac{p(Z_2|Z_1)}{q(Z_2|Z_1,X)}...\frac{p(Z_{t_{max}}|Z_{t_{max}-1})}{q(Z_{t_{max}}|Z_{t_{max}-1},X)}q(Z_1|X)q(Z_2|Z_1,X)...q(Z_{t_{max}})dZ\end{aligned}$$

We may now convert this integral to an expectation with respect to $q(Z|X) = q(Z_1|X)q(Z_2|Z_1,X)...q(Z_{t_{max}}|Z_{t_{max}-1},X)$:

$$= \log \mathop{\mathbb{E}}_{Z\sim q(Z|X)} p(X|Z)\frac{p(Z_1)}{q(Z_1|X)}\frac{p(Z_2|Z_1)}{q(Z_2|Z_1,X)}\cdots\frac{p(Z_{t_{max}}|Z_{t_{max}-1})}{q(Z_{t_{max}}|Z_{t_{max}-1},X)}$$

By Jensen's inequality,

$$\begin{aligned}
&>= \mathop{\mathbb{E}}_{Z\sim q(Z|X)} \log\left\{p(X|Z)\frac{p(Z_1)}{q(Z_1|X)}\frac{p(Z_2|Z_1)}{q(Z_2|Z_1,X)}\cdots\frac{p(Z_{t_{max}}|Z_{t_{max}-1})}{q(Z_{t_{max}}|Z_{t_{max}-1},X)}\right\} \\
&= \mathop{\mathbb{E}}_{Z\sim q(Z|X)}\left[\log p(X|Z) + \log\frac{p(Z_1)}{q(Z_1|X)} + \log\frac{p(Z_2|Z_1)}{q(Z_2|Z_1,X)} + ... + \log\frac{p(Z_{t_{max}}|Z_{t_{max}-1})}{q(Z_{t_{max}}|Z_{t_{max}-1},X)}\right] \\
&= \mathop{\mathbb{E}}_{Z\sim q(Z|X)} \log p(X|Z) - \mathop{\mathbb{E}}_{Z_1\sim q(Z_1|X)} \log\frac{q(Z_1|X)}{p(Z_1)} - \mathop{\mathbb{E}}_{\substack{Z_1\sim q(Z_1|X)\\ Z_2\sim q(Z_2|Z_1,X)}} \log\frac{q(Z_2|Z_1,X)}{p(Z_2|Z_1)} - ... \\
&\qquad\qquad - \mathop{\mathbb{E}}_{\substack{Z_{t_{max}-1}\sim q(Z_{t_{max}-1}|Z_{t_{max}-2},X)\\ Z_{t_{max}}\sim q(Z_{t_{max}}|Z_{t_{max}-1},X)}} \log\frac{q(Z_{t_{max}}|Z_{t_{max}-1},X)}{p(Z_{t_{max}}|Z_{t_{max}-1})}
\end{aligned}$$

Realizing that the expectations $\mathbb{E}\log\frac{q}{p}$ are KL divergences gives us an objective we can optimize:

$$\begin{aligned}
\mathcal{L}(\theta,\phi;X) &= \mathop{\mathbb{E}}_{Z\sim q_\phi(Z|X)} \log p_\theta(X|Z) \\
&\quad - D_{KL}(q_\phi(Z_1|X)||p_\theta(Z_1)) - \sum_{t=2}^{t_{max}} \mathop{\mathbb{E}}_{Z_{t-1}\sim q(Z_{t-1}|Z_{t-2},X)} D_{KL}(q_\phi(Z_t|Z_{t-1},X)||p_\theta(Z_t|Z_{t-1})) \\
&\leq \log p_\theta(X)
\end{aligned}$$

This lower bound lies below the true log-probability by an additive term of $D_{KL}(q(Z|X)||p(Z|X))$ [10]. As the variational approximation $q(Z|X)$ improves (i.e., approaches the true posterior), this lower bound approaches the true log-likelihood of the data.

This bound would hold for any function $p_\theta(Z_t|Z_{t-1})$. Our model factors this general transition function:

$$p_\theta(Z_t|Z_{t-1}) = \prod_{i=1}^{k} p_\theta(z_t^i|z_{t-1}^i)$$

which corresponds to a hidden Markov model with multiple Markov chains running in parallel in the latent space.

Appendix C: Details on datasets

Moving MNIST

This dataset consists of two digits from the MNIST dataset bouncing in a 64x64 pixel frame. Each digit is on a separate plane of the input (i.e., one is red and the other is green). The digits have randomized starting location and velocity vector for each sequence, but their motion is deterministic over the course of the sequence and the digits do not interact.

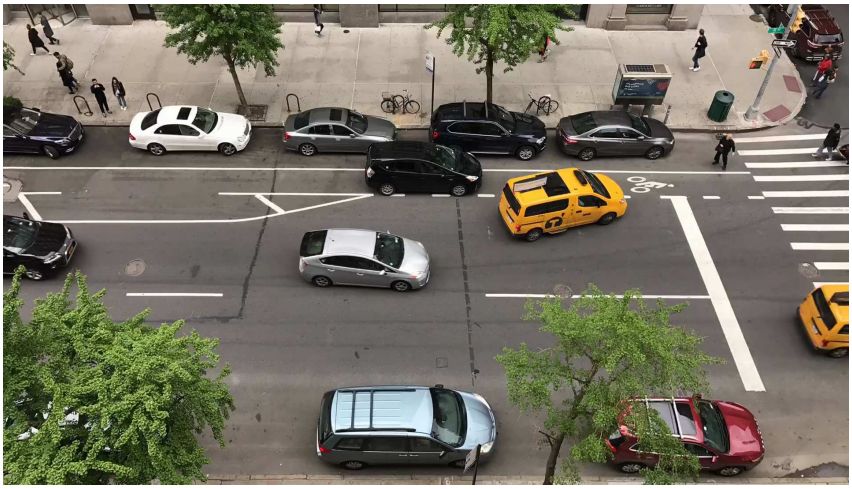


Figure 4: An example image from the collected 5th Avenue dataset of urban videos. These videos include much of the complexity of the real world, including textures, lighting, and uncertain dynamics, while remaining simple enough to model for several frames with a neural network.

5th Avenue

The 5th Avenue dataset has greater complexity in its visuals and its dynamics than moving MNIST, but was designed to be simple enough to model with some fidelity using contemporary techniques. It consists of around 20 hours of video sampled at 2 frames per second. The videos were recorded from the 5th floor of a building overlooking 5th Avenue in Manhattan and show the busy street scene below including pedestrians and passing cars. Each video was recorded with a fixed camera position; between videos the camera position is nearly the same but may vary slightly. The data includes global variations such as time of day and weather. It was recorded on an iPhone 7 at 1080p resolution, though in our experiments we resize it to 64x64. A representative example image is shown in Figure 4.

A Framework for the Quantitative Evaluation of Disentangled Representations

Cian Eastwood
School of Informatics
University of Edinburgh, UK
c.eastwood@ed.ac.uk

Christopher K. I. Williams
School of Informatics
University of Edinburgh, UK
and Alan Turing Institute, London, UK
ckiw@inf.ed.ac.uk

Abstract

Recent AI research has emphasised the importance of learning *disentangled* representations of the explanatory factors behind data. Despite the growing interest in models which can learn such representations, visual inspection remains the standard evaluation metric. While various desiderata have been implied in recent definitions, it is currently unclear what exactly makes one disentangled representation better than another. In this work we propose a framework for quantitatively evaluating disentangled representations. Three criteria are *explicitly* defined and *quantified* to elucidate the quality of learnt representations and compare models on an equal basis. Experiments with the recent InfoGAN model [3] for learning disentangled representations illustrate the appropriateness of the framework and provide a baseline for future work.

1 Introduction

To gain a conceptual understanding of our world, models must first learn to understand the factorial structure of low-level sensory input without supervision [1, 11, 9]. As argued in several notable works [6, 1, 9, 3], this understanding can only be gained if the model learns to *disentangle* the underlying explanatory factors hidden in *unlabelled* input.

A disentangled representation is generally described as one which separates the factors of variation, explicitly representing the important attributes of the data [6, 1, 10, 9, 3]. For example, given an image dataset of human faces, a disentangled representation may consist of separate dimensions (or features) for the face size, hairstyle, eye colour, facial expression, etc. Despite the expanding literature on models which seek to learn such disentangled representations [6, 13, 4, 10, 3, 9], visual inspection remains the standard evaluation metric. While the work of Higgins et al. [9] partially addresses this issue (as discussed in section 2) and various definitions have implied additional desiderata like interpretability and invariance [6, 1, 13, 5, 10, 3], current research generally lacks a clear metric for quantitatively evaluating and comparing disentangled representations.

In this work we propose a framework to quantitatively evaluate disentangled representations. To elucidate the quality of learnt representations and compare models on an equal basis, desiderata of disentangled representations are *explicitly* defined and *quantified*. These desiderata help define the disentangled representations which we seek and remove the need for a subjective visual evaluation by a human arbiter. To illustrate the appropriateness of this framework, we use it to quantitatively evaluate the representations learned by information maximizing generative adversarial networks (InfoGAN) [3].

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

2 Theoretical Framework

Models for disentangled factor learning seek a compact D-dimensional data representation or *code* which consists of disentangled and interpretable latent variables. For graphics-generated data, the K-dimensional generative factors $\boldsymbol{z}$ are designed to be an ideal such representation. Thus, if $K = D$, the ideal disentangled code $\boldsymbol{c}^*$ should be some permutation of $\boldsymbol{z}$. That is, $\boldsymbol{c}^* = f^*(\boldsymbol{z})$, where the ideal mapping f^* is a generalised permutation matrix (monomial matrix[1]). As $f^{*-1} = f^{*T}$, we can instead write $\boldsymbol{z} = f^{*T}(\boldsymbol{c}^*)$ to interpret the monomial matrix f^{*T} as a regressor which predicts $\boldsymbol{z}$ from $\boldsymbol{c}^*$. If $D > K$, f^{*T} will be a monomial matrix over K of the code variables, with zero contribution from the remaining code variables. For notational simplicity, we now use f^* (rather than f^{*T}) to denote this ideal regressor. Thus, we can quantitatively evaluate the codes learned by a given model M using the following steps:

1. Train M on a synthetic dataset with generative factors $\boldsymbol{z}$
2. Retrieve $\boldsymbol{c}$ for each sample $\boldsymbol{x}$ in the dataset ($\boldsymbol{c} = M(\boldsymbol{x})$)
3. Train regressor f to predict $\boldsymbol{z}$ given $\boldsymbol{c}$ ($\hat{\boldsymbol{z}} = f(\boldsymbol{c})$)
4. Quantify f's deviation from f^* and the prediction error

We now detail the proposed evaluation metrics, i.e., steps 3 and 4. We train K regressors to predict the value of K generative factors. The regressor f_j predicts z_j given $\boldsymbol{c}$, that is, it learns a mapping $f_j(\boldsymbol{c}) : \mathbb{R}^D \rightarrow \mathbb{R}^1$. We begin with linear regressors and encourage a *sparse* mapping between $\boldsymbol{c}$ and $\boldsymbol{z}$ with an ℓ_1 regularisation penalty (lasso regressors). With the inputs and targets normalised to have zero mean and unit variance, the magnitude of the resulting regression weights rank the learnt code variables $c_0, \ldots, c_{D-1}$ in order of relative importance to the prediction. That is, they reveal which code variables capture information about a given generative factor. Thus, we define the matrix of relative importances R as $R_{ij} = |W_{ij}|$ for linear regression, where R_{ij} denotes the relative importance of c_i in predicting z_j and $|W_{ij}|$ denotes the magnitude of the weight used to scale c_i in predicting z_j. This allows us to explicitly define and quantify three criteria of disentangled representations (or *codes*) which are implicit in recent definitions [6, 1, 13, 5, 10, 9, 3], namely *disentanglement*, *informativeness* and *completeness*.

Disentanglement. The degree to which a representation factorises or *disentangles* the underlying factors of variation, with each variable (or feature) capturing at most one generative factor. Disentanglement implies invariance to all but one generative factor and distinguishes genuinely disentangled representations from those that are just statistically independent[2]. The disentanglement score D_i of code variable c_i is quantified by $D_i = 1 - H_K(P_{i*})$, where $H_K(P_{i*}) = -\sum_{j=0}^{K-1} P_{ij} \log_K P_{ij}$ denotes the entropy and $P_{ij} = R_{ij} / \sum_{j=0}^{K-1} R_{ij}$ denotes the 'probability' of c_i being important for predicting z_j. If c_i is important for predicting a single generative factor, the score will be 1. If c_i is equally important for predicting all generative factors, the score will be 0. D_i can be visualised by examining row i of the Hinton diagrams as in Figure 2.

Informativeness. The amount of information that a representation captures about the underlying factors of variation. To be useful for natural tasks which require knowledge of the important attributes of the data (e.g. object recognition), representations must ultimately capture information about the underlying factors of variation [1, 3]. The informativeness of a representation or *code* $\boldsymbol{c}$ about a given generative factor z_j is quantified by the prediction error $E(z_j, \hat{z}_j)$ (averaged over the dataset), where E is an appropriate error function and $\hat{z}_j = f_j(\boldsymbol{c})$.

Completeness. The degree to which the underlying factors of variation are captured with a representation of equal dimensionality. The completeness score C_j in capturing generative factor z_j is quantified by $C_j = 1 - H_D(\tilde{P}_{*j})$, where $H_D(\tilde{P}_{*j}) = -\sum_{i=0}^{D-1} \tilde{P}_{ij} \log_D \tilde{P}_{ij}$ denotes the entropy and $\tilde{P}_{ij} = R_{ij} / \sum_{i=0}^{D-1} R_{ij}$ denotes the 'probability' of c_i being important for predicting z_j. If a single code variable contributes to z_j's prediction, the score will be 1 (complete). If all code variables equally contribute to z_j's prediction, the score will be 0 (maximally overcomplete). C_j can be visualised by examining column j of the Hinton diagrams as in Figure 2.

[1] A matrix is monomial if there is exactly one non-zero element in each row and column. If the non-zero elements have value 1 the matrix is a permutation matrix.

[2] E.g. in a Gaussian factor analysis model we may well obtain a rotated version of the true generative factors due to the spherical symmetry of the prior (the rotation of factors problem).

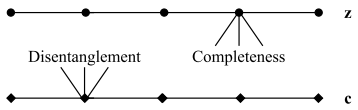


Figure 1: **Visualising disentanglement and completeness.** A one-to-one mapping is ideal. We can quantify the deviation from this ideal mapping using the disentanglement and completeness scores.

Together, the degree of disentanglement and completeness quantify the deviation from the ideal regressor f^*. While disentanglement quantifies the number of generative factors captured by a given code variable, completeness quantifies the number of code variables which capture a given generative factor. Figure 1 illustrates this idea for $K = D$. While the ideal code would be able to explicitly represent each generative factor with a single variable, models with generic priors cannot be expected to learn such complete codes (representations). For example, generative factors which are drawn from a distribution on a circle cannot be accurately captured by single code variables on which unwrapped prior distributions are imposed. Thus, with generic priors like the standard normal, information about such generative factors may be non-linearly encoded across multiple code variables. Empirical results in Appendix B, Appendix C and [9] support this idea, with several code variables resembling non-linear functions (like the sine and cosine) of the object azimuth. This motivates the use of non-linear regressors. We use random forest regressors due to their inbuilt ability to determine the relative importance of each feature to a given prediction, thus allowing us to quantify the degree of disentanglement and completeness as before by directly specifying the matrix of relative importances R. Random forests average the predictions and feature importances from each decision tree in the ensemble. The number of times a tree chooses to split on a particular input variable determines its importance to the prediction. Thus, the relative importance of each input variable c_i is given by the number of cases split on c_i over the total number of splits [2]. In addition to quantifying the deviation from f^*, the disentanglement and completeness scores clearly expose any disentangling that is done by the non-linear regressor itself. As performance generally improves with the number of trees n in the ensemble, we fix $n = 10$ and fit all other parameters (and hyperparameters) to a validation set.

Related Work. Higgins et al. [9] propose a metric to quantify the degree of disentanglement achieved by different models. In this work, an additional dataset of 'factor changes' is generated by taking the absolute difference between two latent representations corresponding to images which differ only by a change in a single generative factor. Given these changes in latent space, a linear classifier is trained to predict which generating factor caused the change between the images, with the classification accuracy quantifying the degree of disentanglement. While this metric quantifies the degree of disentanglement (latent variables must be primarily perturbed by changes in a single generative factor to achieve high classification accuracy), it does not quantify the amount of information captured about the generative factors or the completeness of the representation. By quantifying these additional criteria, our simple metrics provide a more thorough evaluation of the disentangled representations learned by a given model, without the need to generate an additional dataset.

3 Results

To demonstrate the appropriateness of the framework, we train InfoGAN [3] with 6 latent variables on a synthetic dataset with 5 generative factors, quantitatively comparing the learned representations or *codes* to those learned by PCA (with 6 latent variables). As disentangled codes should enable a model to generalise its knowledge beyond the training distribution by recombining previously-learnt factors (i.e. perform zero-shot inference) [1, 9], we also compare the codes learned by each model on data containing unseen factor combinations. Note that the (latent) variables in InfoGAN's learned *code* are termed 'latent codes' in [3] while those in PCA's code are termed principal components. Further details on the dataset(s) and InfoGAN model are provided in Appendix A.

Tables 1 and 2 present the results for the lasso and random forest regressors respectively. As each target is normalised to have a standard deviation of 1, the root-mean-square error (RMSE) in predicting each target is naturally normalised relative to the constant regressor which guesses the expected value of the targets. Hence, we report the normalised root-mean-square error (NRMSE) in these tables. Table 1a presents the test set NRMSE for images containing factor combinations similar to those on which the models were trained. The representation or *code* learned by InfoGAN ($\boldsymbol{c}-\mathrm{InfoGAN}$) clearly outperforms the code learned by PCA ($\boldsymbol{c}-\mathrm{PCA}$) in predicting each generative factor. That is, it is far more *informative*. It is worth noting the significantly higher error in predicting

(a) **Test set NRMSE**

Code	z_0	z_1	z_2	z_3	z_4	Avg.
PCA	0.99	0.72	0.42	0.47	0.46	0.59
InfoGAN	0.50	0.09	0.09	0.13	0.09	0.18

(b) **Disentanglement**

Code	c_0	c_1	c_2	c_3	c_4	c_5	W. Avg.
PCA	0.16	0.38	0.42	0.14	0.63	0.16	0.20
InfoGAN	0.83	0.64	0.65	0.93	0.80	0.80	0.77

(c) **Zero-shot NRMSE**

Code	z_0	z_1	z_2	z_3	z_4	Avg.
PCA	1.01	1.78	0.77	0.91	0.91	1.05
InfoGAN	0.49	0.20	0.35	0.21	0.16	0.28

(d) **Completeness**

Code	z_0	z_1	z_2	z_3	z_4	Avg.
PCA	0.83	0.54	0.14	0.20	0.19	0.38
InfoGAN	0.56	0.85	0.80	0.63	0.78	0.72

Table 1: **Lasso regression results**. **(a)** Test set NRMSE on images containing factor combinations similar to those on which the models were trained **(b)** Disentanglement scores for each variable in $\boldsymbol{c}-\mathrm{InfoGAN}$ and $\boldsymbol{c}-\mathrm{PCA}$. 'W. Avg.' abbreviates 'weighted average'. **(c)** NRMSE on images containing unseen factor combinations. **(d)** Completeness scores for each generative factor.

(a) **Test set NRMSE**

Code	z_0	z_1	z_2	z_3	z_4	Avg.
PCA	0.65	0.41	0.27	0.34	0.34	0.40
InfoGAN	0.29	0.06	0.06	0.08	0.07	0.11

(b) **Disentanglement**

Code	c_0	c_1	c_2	c_3	c_4	c_5	W. Avg.
PCA	0.19	0.38	0.47	0.53	0.87	0.15	0.35
InfoGAN	0.96	0.90	0.91	0.93	0.89	0.96	0.93

(c) **Zero-shot NRMSE**

Code	z_0	z_1	z_2	z_3	z_4	Avg.
PCA	0.71	1.12	0.86	0.87	0.81	0.87
InfoGAN	0.30	0.17	0.30	0.19	0.14	0.22

(d) **Completeness**

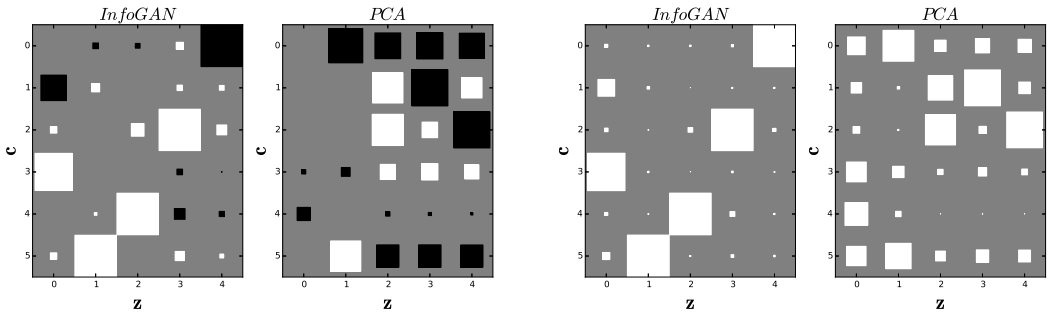
Code	z_0	z_1	z_2	z_3	z_4	Avg.
PCA	0.11	0.45	0.37	0.49	0.49	0.38
InfoGAN	0.63	0.97	0.95	0.92	0.96	0.89

Table 2: **Random forest regression results**. Caption of Table 1 applies.

the azimuth (z_0) from $\boldsymbol{c}-\mathrm{InfoGAN}$. This can be attributed to: (i) the low-capacity linear regressor being unable to extract the information encoded in $\boldsymbol{c}-\mathrm{InfoGAN}$ about the azimuth (indicated by much lower azimuth prediction error of the non-linear regressor in Table 2a); (ii) InfoGAN struggles to capture enough information about the azimuth in $\boldsymbol{c}-\mathrm{InfoGAN}$ (indicated by the relatively high error in predicting the azimuth with both regressors). Comparing the results of the linear regressor in Table 1a with those of the non-linear regressor in Table 2a, we see that both codes better predict the generative factors with increased capacity—especially $\boldsymbol{c}-\mathrm{PCA}$.

Tables 1b and 2b present the disentanglement scores for the lasso and random forest regressors respectively. To prevent the disentanglement scores of redundant or unimportant code variables from skewing the average, relative code variable importance $R_i = \sum_j R_{ij} / \sum_{i,j} R_{ij}$ is used to construct a weighted average. With both regressors, the variables in $\boldsymbol{c}-\mathrm{InfoGAN}$ achieve a much higher disentanglement score than those in $\boldsymbol{c}-\mathrm{PCA}$, with each variable in $\boldsymbol{c}-\mathrm{InfoGAN}$ closer to capturing a single generative factor. That is, $\boldsymbol{c}-\mathrm{InfoGAN}$ is more *disentangled*. The high disentanglement scores of $\boldsymbol{c}-\mathrm{InfoGAN}$ in Table 2b confirm that the low NRMSEs in Table 2a were not due to any substantial disentangling done by the (non-linear) random forest regressor itself. The same cannot be said for $\boldsymbol{c}-\mathrm{PCA}$, with its low disentanglement scores in Table 2b revealing that this 'tangled' code is in fact disentangled by the non-linear regressor itself to achieve lower NRMSEs. Figure 2 helps to visualise the disentanglement and identify the generative factors captured by each code variable. For example, comparing $c_0-\mathrm{PCA}$ and $c_0-\mathrm{InfoGAN}$ in figures 2a and 2b (the first rows), it is clear that $c_0-\mathrm{PCA}$ captures information about each generative factor while $c_0-\mathrm{InfoGAN}$ (almost) solely captures information about z_4. The lack of disentanglement within $\boldsymbol{c}-\mathrm{PCA}$ indicates that PCA is unable to separate the factors of variation ($\boldsymbol{z}$) in the data, ultimately preventing it from generalising to data with unseen factor combinations. This is illustrated by the high zero-shot NRMSE of $\boldsymbol{c}-\mathrm{PCA}$ in tables 1c and 2c, with $\boldsymbol{c}-\mathrm{PCA}$ even outperformed by the constant regressor in predicting z_1. In contrast, $\boldsymbol{c}-\mathrm{InfoGAN}$ predicts the value of unseen factor combinations reasonably well.

Tables 1d and 2d present the completeness scores for the lasso and random forest regressors respectively. The high completeness scores of $\boldsymbol{c}-\mathrm{InfoGAN}$ reveal that it captures each generative factor with approximately one code variable. That is, they show that $\boldsymbol{c}-\mathrm{InfoGAN}$ is almost complete. In contrast, the low scores of $\boldsymbol{c}-\mathrm{PCA}$ reveal that is it severely overcomplete, using several code variables to capture each generative factor. Again, Figure 2 helps to identify the generative factors captured by a given code variable and visualise the completeness. Figure 2 also helps to explain the low completeness score (overcompleteness) of $\boldsymbol{c}-\mathrm{InfoGAN}$ in predicting z_0, with z_0 clearly captured



(a) **Lasso weight magnitudes** (b) **Random forest feature importances**

Figure 2: **Visualising disentanglement and completeness within learnt representations.** Positive weights/importances are represented by a white square and negative by a black square, while size indicates the magnitude. Row i illustrates the importance of c_i to each prediction and thus the disentanglement. Column j illustrates the relative importance of each code variable for predicting z_j and thus the completeness. Ideally, there would be a single large weight in K rows and each column.

by a combination of $c_1-\mathrm{InfoGAN}$ and $c_3-\mathrm{InfoGAN}$. However, this is still less overcomplete than $\boldsymbol{c}-\mathrm{PCA}$, with each of its constituent variables capturing information about z_0 (see Figure 2b).

Conclusion. In this work we have presented a framework for quantitatively evaluating the disentangled representations learned by different models. The quality of learnt representations is elucidated through the explicit definition and quantification of three criteria: disentanglement, informativeness and completeness. In addition, the quantitative results of InfoGAN illustrate the framework's appropriateness and provide a baseline for future models which seek to learn disentangled representations without supervision. While we have focused on image data in this work, we hope that future work will explore the applicability of the framework to other types of synthetic data.

Acknowledgements. We would like to thank Pol Moreno for generating the dataset(s) and insightful comments. We would also like to thank Akash Srivastava and Andrew Brock for helpful discussions. The work of CW is supported in part by EPSRC grant EP/N510129/1.

References

[1] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE TPAMI 2013*, 35(8):1798–1828.

[2] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[3] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS 2016*, pages 2172–2180.

[4] B. Cheung, J. A. Livezey, A. K. Bansal, and B. A. Olshausen. Discovering hidden factors of variation in deep networks. *arXiv preprint arXiv:1412.6583*, 2014.

[5] T. S. Cohen and M. Welling. Transformation properties of learned visual representations. *arXiv preprint arXiv:1412.7659*, 2014.

[6] G. Desjardins, A. Courville, and Y. Bengio. Disentangling factors of variation via generative entangling. *arXiv preprint arXiv:1210.5474*, 2012.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS 2014*, pages 2672–2680.

[8] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of Wasserstein GANs. *arXiv preprint arXiv:1704.00028*, 2017.

[9] I. Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, C. Blundell, S. Mohamed, and A. Lerchner. Early visual concept learning with unsupervised deep learning. *arXiv preprint arXiv:1606.05579*, 2016.

[10] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *NIPS 2015*, pages 2539–2547.

[11] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, pages 1–101, 2016.

[12] P. Moreno, C. K. I. Williams, C. Nash, and P. Kohli. Overcoming occlusion with inverse graphics. In *ECCV Geometry Meets Deep Learning Workshop 2016*, pages 170–185. Springer.

[13] S. Reed, K. Sohn, Y. Zhang, and H. Lee. Learning to disentangle factors of variation with manifold interaction. In *ICML 2014*, pages 1431–1439.

A Experimental setup

A.1 Data

The graphics renderer described in [12] was employed to generate 200,000 images of an object (teapot) with varying pose and colour (see Figure 3a). For simplicity, the camera is centred on the object, the scene background is removed and additional generative factors (shape and lighting) are held constant. Each generative factor is independently sampled from its respective uniform distribution: azimuth(z_0) $\sim U[0, 2\pi]$, elevation(z_1) $\sim U[0, \pi/2]$, red(z_2) $\sim U[0, 1]$, green(z_3) $\sim U[0, 1]$, blue(z_4) $\sim U[0, 1]$. In line with recent architectures, we set the image dimensions to be $64 \times 64 \times 3$.

We use the ground-truth values of the generative factors to create two different data distributions. More specifically, we isolate all images whose generative factor values lie in a particular range to create a 'gap' in the original dataset. This gap then serves as our zero-shot data containing unseen factor combinations. Informally, the images in this gap can be described as 'red' teapots from 'above'. Formally, the generative factors of these images satisfy the following condition: $z_2 > (z_3 + 0.15)$ and $z_2 > (z_4 + 0.15)$ and $z_1 > \frac{\pi}{4}$. This dataset contained approximately 20,000 images, with (extreme) samples given in Figure 3b.

(a) **Seen factor combinations**

(b) **Unseen factor combinations**

Figure 3: **Data samples.** (a) Examples of images (and corresponding generative factor combinations) on which the models are trained. (b) Examples of images in the 'gap' that was created containing unseen factor combinations.

A.2 Model

Extending the GAN of Goodfellow et al. [7], InfoGAN [3] splits the generator input into two parts; 'incompressible noise' and 'latent codes' which target salient semantic features of the data. The manner in which the generator may use these latent codes is constrained by adding a regularisation term to the GAN objective, representing the mutual information between the latent codes and generated images. For stability, we use the training objective of the improved Wasserstein GAN (IWGAN) [8] and the 64×64 ResNet architecture described in the open source implementation of Gulrajani et al. [8][3]. We modify this implementation to add InfoGAN's variational regularisation of mutual information to the IWGAN training objective, splitting the generator input into noise and latent code components before implementing the auxiliary network Q as in [3]. The network Q parametrises the approximate posterior over latent codes $Q(\boldsymbol{c}|\boldsymbol{x})$, with $Q(\boldsymbol{x})$ returning a mean and standard deviation for each continuous (normal) latent code in the factorised posterior $Q(\boldsymbol{c}|\boldsymbol{x})$. Thus, to retrieve the (most likely) representation or code $\boldsymbol{c}$ for a given image $\boldsymbol{x}$, we simply take the means returned by $Q(\boldsymbol{x})$. Q shares all convolutional layers with the discriminator or 'critic' D, each adding their own final output layer. All hyperparameters of the IWGAN implementation remain unchanged while we found setting the mutual information coefficient $\lambda = 8$ to be sufficient, ensuring that the mutual information penalty was on the same scale as the unbounded WGAN objectives. For all experiments, we use 6 continuous latent codes and 128 noise variables resulting in a generator input with dimension 134. For illustrative purposes, we show the best of 10 random runs as we found InfoGAN to be quite sensitive to random initialisation. Further details on the architecture, hyperparameters and combined training objective are provided in our open-source implementation, to be made publicly available on acceptance of this paper.

[3] `https://github.com/igul222/improved_wgan_training`

B Generative factors vs. latent codes with generic priors

Figure 4 plots each generative factor against the corresponding 'most important' InfoGAN code variable, as indicated by the lasso regression weight magnitudes and random forest feature importances. Information about each unwrapped generative factor (z_1, z_2, z_3, z_4) is linearly-encoded in single code variables (c_5, c_4, c_2, c_0 respectively). In contrast, distinct information about the azimuth (z_0) is non-linearly encoded across c_1 and c_3, reinforcing the argument that models with generic priors cannot be expected to learn the most complete and explicit representation of topologically distinct factors of variation. Furthermore, when InfoGAN was retrained instead with 10 code variables, it used three of these to capture the azimuth (see Figure 5a), each resembling (scaled) sine and cosine functions (see Figure 5b).

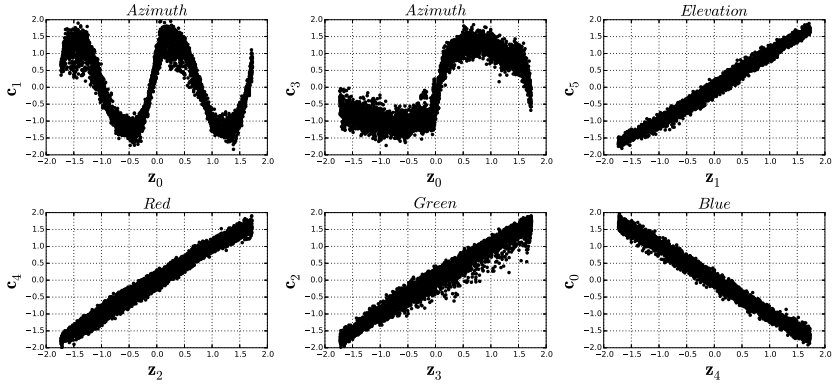
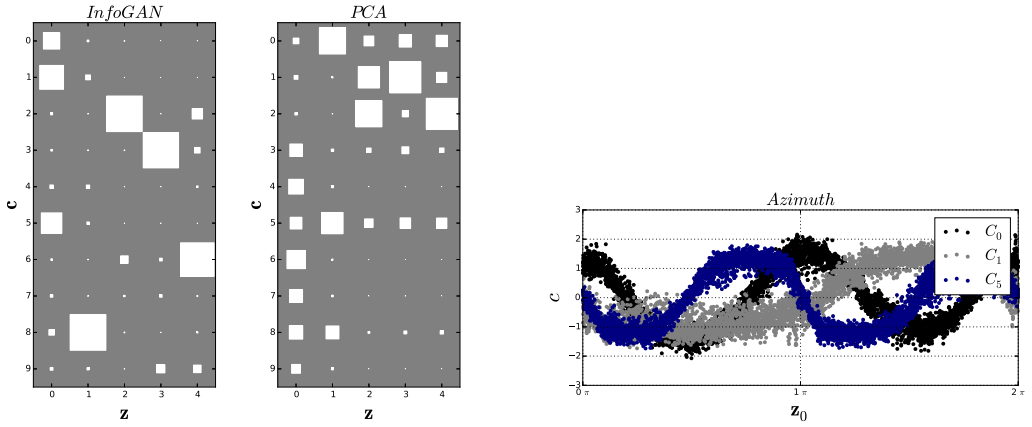


Figure 4: **Generative factors vs. 'most important' InfoGAN code variables.**

C $\mathbf{K = 5, D = 10}$

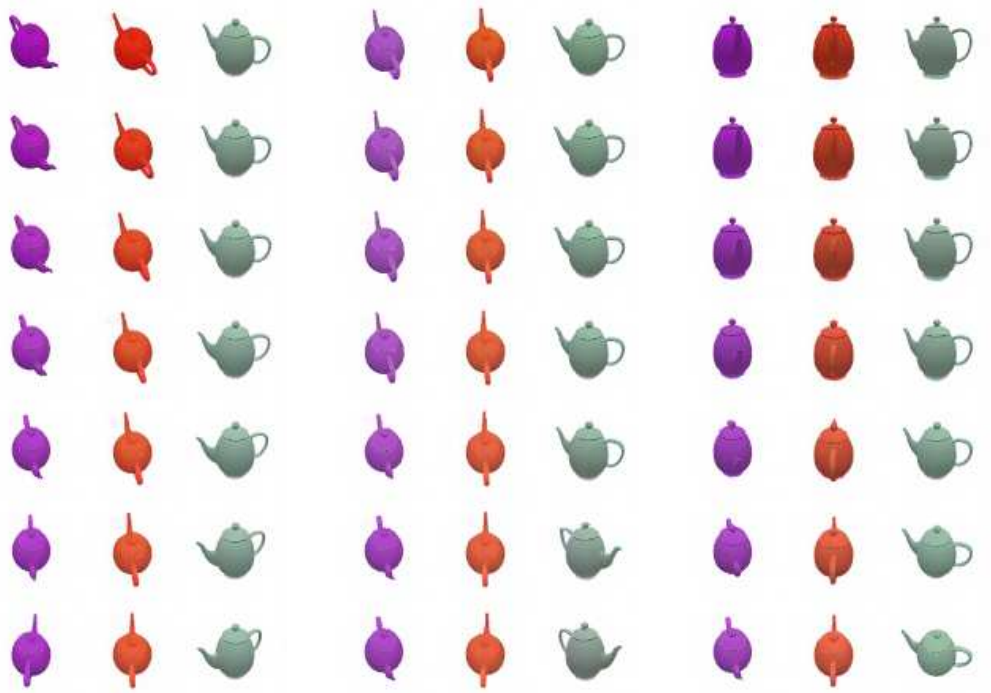
As shown in Figure 5a, several redundant code variables (c_4, c_7, c_9) enable a high degree of completeness in $\boldsymbol{c}-$InfoGAN. Although $\boldsymbol{c}-$InfoGAN captures z_0 with 3 code variables, $\boldsymbol{c}-$PCA uses a least 7. Figure 5b shows that the 3 InfoGAN code variables which capture z_0 (azimuth) resemble scaled versions of sine and cosine functions. While the regression results for InfoGAN with 10 latent codes were inferior, Figure 5a bodes well for InfoGAN's ability to learn disentangled and interpretable codes without any knowledge about the number of underlying factors of variation (i.e. when trained with excessive latent codes). We note that further hyperparameter searches and random runs may have yielded better results.



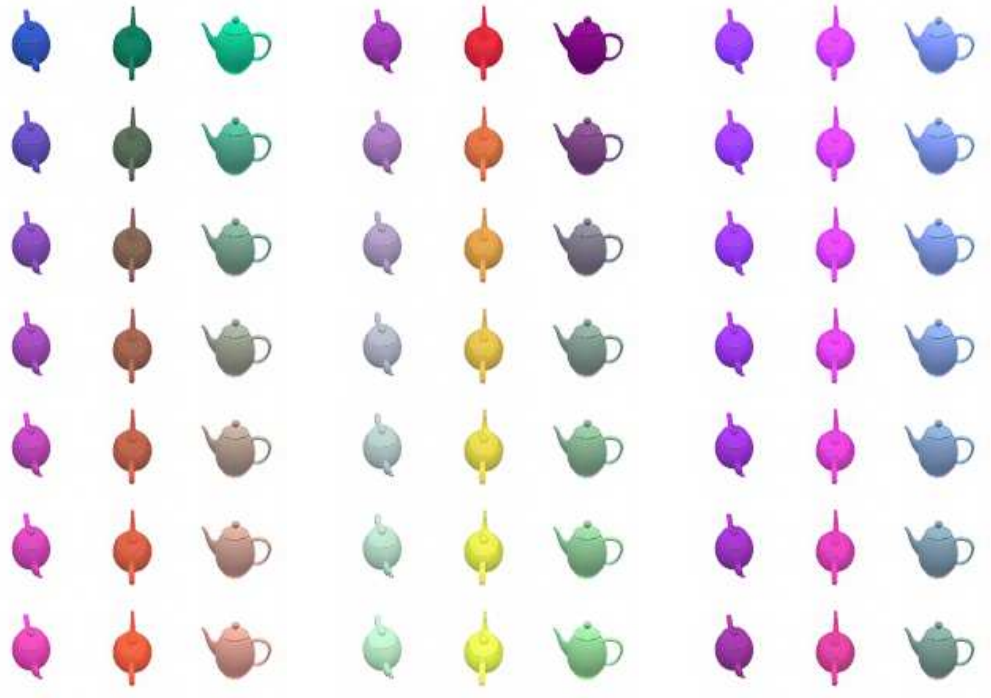
(a) **Random forest feature importances** (b) **Learnt codes vs. generative factors**

Figure 5: **Visualising the degree of disentanglement and completeness within learnt representations.**

D Visually assessing disentanglement



(a) **Azimuth(c_1)** (b) **Azimuth(c_3)** (c) **Elevation(c_5)**



(d) **Red(c_4)** (e) **Green(c_2)** (f) **Blue(c_0)**

Figure 6: **Manipulating the latent codes.** Each subfigure column represents a different random sample (or initialisation) of $\boldsymbol{c}$. For each random sample, c_i is varied from -1 (bottom) to 1 (top) to show the effect on generated images. There appears to be a high degree of disentanglement as each subfigure contains a single type of semantic variation.

Disentanglement by penalizing correlation

Mikael Kågebäck & Olof Mogren
Chalmers University of Technology, Sweden
[kageback,mogren]@chalmers.se

Abstract

An important reason for the success of deep neural networks this is their capability to automatically learn representations of data in levels of abstraction, increasingly disentangling the data as the internal transformations are applied. In this paper we propose a novel regularization method that actively penalize covariance between dimensions of the hidden layers in a network, driving the model towards a more disentangled solution. This makes the network learn linearly uncorrelated representations which increases interpretability while obtaining good results on a number of tasks, as demonstrated by our experimental evaluation. Further, the proposed technique effectively disables superfluous dimensions, compressing the representation to the dimensionality of the underlying data. Our approach is computationally cheap and can be applied as a regularizer to any gradient-based learning model.

1 Introduction

A good data representation should uncover underlying factors in the data while being useful for some task. Deep networks learn representations of increasing abstraction, disentangling the causes of variation in the underlying data (Bengio et al., 2013). Formal definitions of disentanglement are lacking, although Ver Steeg & Galstyan (2015); Achille & Soatto (2017) both use the total correlation as a measure of disentanglement. Inspired by this, we consider a simpler objective: a representation disentangles the data well when its components do not correlate, and we explore the effects of penalizing this linear dependence betweeen different dimensions in the representation.

We propose L_Σ regularization, a novel regularization scheme that penalizes the correlation between the dimensions of the learned representations. The approach is very versatile and can be applied to any gradient-based machine learning model that learns its own distributed vector representations. Compared to previous work on learning independent nonlinear representations our approach is simpler, and does not impose restrictions on the model used. The approach strongly encourages the model to find the dimensionality of the data, something that is verified by the experimental evaluation. This can be of great utility when pruning a network, or to decide when a network needs a larger capacity. The disabling of activations in the internal representation can be viewed as (and used for) dimensionality reduction. The proposed approach allows for interpretability of the activations computed in the model, such as isolating specific underlying factors. The solution is computationally cheap, and can be applied without modification to many gradient-based machine learning models that learns distributed representations. Moreover, we present an extensive experimental evaluation on a range of tasks on different data modalities using different model layouts, which shows that the proposed approach disentangles the data well; we do get uncorrelated components in the resulting internal representations, while retaining the performance of the models on their respective task.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

2 Disentanglement by penalizing correlation

We present a novel regularizer based on the covariance of the activations in a neural network layer over a batch of examples. The aim of the regularizer is to penalize the covariance between dimensions in the layer to decrease linear correlation.

Definition The covariance regularization term (L_Σ) for a layer, henceforth referred to as the coding layer, is computed as $L_\Sigma = \frac{1}{p^2}||\mathcal{C}||_1$ where p is the dimensionality of the coding layer, $||\mathcal{C}||_1 = \sum_{i,j=1}^{N} |\mathcal{C}_{ij}|$ is the element wise L1 matrix norm of $\mathcal{C}$, and $\mathcal{C} \in \mathcal{R}^{p \times p}$ is the sample covariance of the activations in the coding layer over N examples $\mathcal{C} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{H} - \mathbf{1}_N\bar{\mathbf{h}})^T(\mathbf{H} - \mathbf{1}_N\bar{\mathbf{h}})$.Further, $\mathbf{H} = [\mathbf{h}_1; ...; \mathbf{h}_N]$ is a matrix of all activations in the batch, $\mathbf{1}_N$ is an N-dimensional column vector of ones, and $\bar{\mathbf{h}}$ is the mean activation.

Usage As L_Σ has the structure of a regularizer, it can be applied to most gradient based models without changing the underlying architecture. In particular, L_Σ is simply computed based on select layers and added to the error function, e.g. $Loss = Error + \lambda L_\Sigma$

3 Experiments

This section describes the experimental evaluation of L_Σ regularization in different settings.

3.1 Evaluation metrics

Mean Absolute Pearson Correlation (MAPC) Pearson correlation report the normalized linear correlation between variables $\in [-1, 1]$. MAPC is the average absolute value of the correlation between all dimensionss, i.e. $\mathbf{MAPC} = (2/(p^2 - p))\sum_{i<j}^{p} |\mathcal{C}_{ij}|/\sqrt{\mathcal{C}_{ii}}\sqrt{\mathcal{C}_{jj}}$
Covariance/Variance Ratio (CVR) Mean absolute Pearson correlation becomes ill defined when the variance of one (or both) of the variables approaches zero. We define a related measure where all variances are summed for each dimension. More precise, the score is computed as: $\mathbf{CVR} = \frac{1}{p^2}\frac{||\mathcal{C}||_1}{\mathbf{tr}(\mathcal{C})}$where $||\mathcal{C}||_1$ is defined as in Sec 2. The intuition behind CVR is simply to measure the fraction of all information that is captured in a linear uncorrelated fashion within the coding layer.
Top d-dimension Variance/total variance (TdV) TdV measure to what degree the total variance is captured inside the variance of the top d dimensions.
90% Utilized Dimensions ($\mathbf{UD}_{90\%}$) The number of dimensions that needs to be kept to retain 90% of the total variance.

3.2 Dimensionality reduction

The purpose of this experiment is to investigate if it is possible to disentangle independent data that has been projected to a higher dimension using a random projection, i.e. we would like to find the principal components of the original data.

The model we employ in this experiment is an auto encoder consisting of a linear $p = 10$ dimensional coding layer and a linear outputlayer. The model is trained using the proposed covariance regularization L_Σ on the coding layer. The data is generated by sampling a $d = 4$ dimensional vector of independent features $z \sim N(0, \Sigma)$, where $\Sigma \in \mathcal{R}^{d \times d}$ is constrained to be non-degenerate and diagonal. However, before the data is fed to the autoencoder it is pushed through a random linear transformation $x = \Omega z$. The goal of the model is to reconstruct properties of z in the coding layer while only having access to x. The model is trained on 10000 iid random samples for 10000 epochs. 9 experiments were performed with different values for the regularization constant λ. The first point on each curve (in Fig 1 and 2) is $\lambda = 0$, i.e. no regularization, followed by 8 points logarithmically spaced between 0.001 and 1. Each experiment is repeated 10 times using a different random projection Ω and the average is reported.

The result of the experiment is reported using all four metrics defined in Sec 3.1. The result in terms of MAPC and CVR is reported in Fig 1. The first thing to notice is that L_Σ consistently lead to lower correlation while incurring less MSE penalty compared to L_1. Further, looking at the MAPC it is interesting to notice that it is optimal for a very small values of L_Σ. This is because higher amounts of L_Σ leads to lowering of the dimensionality of the data, see Fig 2, which in turn yields

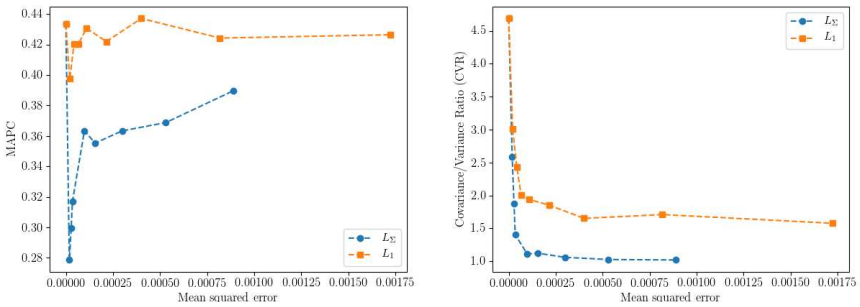


Figure 1: The amount of residual linear correlation after training the model with L_Σ and L_1 regularization respectively, measured in MAPC (left) and CVR (right). The first point on each curve corresponds to $\lambda = 0$, i.e. no regularization, followed by 8 points logarithmically spaced between 0.001 and 1. All scores are averaged over 10 experiments using a different random projection (Ω).

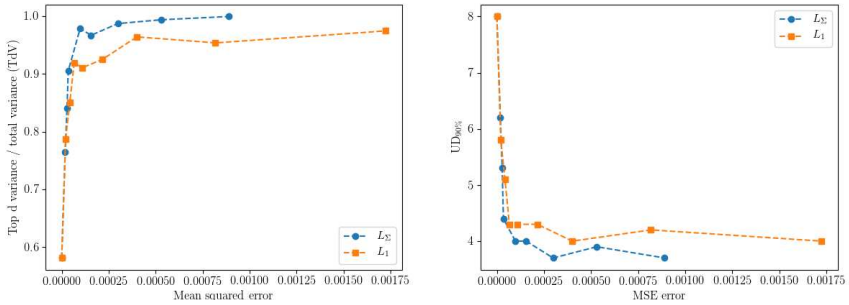


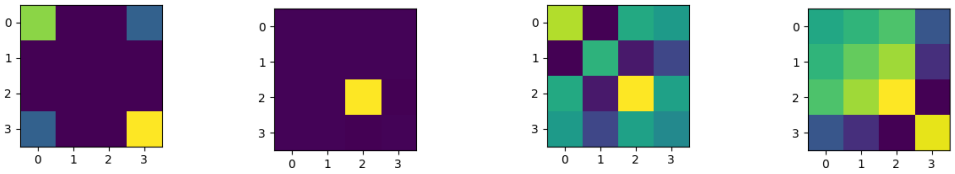
Figure 2: The resulting dimensionality the coding layer after training the model with L_Σ and L_1 regularization respectively, measured in TdV (left) and $\text{UD}_{90\%}$ (right). The first point on each curve corresponds to $\lambda = 0$, i.e. no regularization, followed by 8 points logarithmically spaced between 0.001 and 1.All scores are averaged over 10 experiments using a different random projection (Ω).

unpredictable Pearson correlation scores between these inactivated neurons. However, this effect is compensated for in CVR for which L_Σ quickly converges towards the optimal value of one, which in turn indicates no presence of linear correlation. Turning the attention to dimensionality reduction, Fig 2 shows that L_Σ consistently outperform L_1. Further, looking closer at the TdV score, L_Σ is able to compress the data almost perfectly, i.e. TdV=1, at a very small MSE cost while L_1 struggle even when accepting a much higher MSE cost. Further, the $\text{UD}_{90\%}$ scores again show that L_Σ achieves a higher compression at lower MSE cost. In this instance the underlying data was of 4 dimensions which L_Σ quickly achieves. At higher amounts of L_Σ the dimensionality even occasionally fall to 3, however, this is due to the threshold of 90%.

3.3 Deep network of uncorrelated features

In Sec 3.2 we showed that we can learn a minimal orthogonal representation of data that is generated to ensure that each dimension is independent. However, in reality it is not always possible to encode the necessary information, to solve the problem at hand, in an uncorrelated coding layer. However, using a deep network it should be possible to learn such a nonlinear transformation that enables uncorrelated features in higher layers. To test this in practice on a problem that has this property but still is small enough to easily understand we turn to the XOR problem.

It is well known that the XOR problem can be solved by a neural network of one hidden layer consisting of a minimum of two units. However, instead of providing this minimal structure we would like the network to discover it by itself during training. Hence, the model used is intentionally over-specified consisting of two hidden layers of four logistic units each followed by a one dimensional logistic output layer. The model was trained on XOR examples, e.g. [1,0]=1, in a random order until convergence with L_Σ applied to both hidden layers with $\lambda = 0.2$.



(a) Layer 1, L_Σ (b) Layer 2, L_Σ (c) Layer 1, baseline (d) Layer 2, baseline

Figure 3: Covariance matrices of the hidden layers when trained while applying L_Σ regularization (a and b) to solve the XOR problem. Layer one has learned to utilize unit zero and three while keeping the rest constant, and in layer two only unit two is utilized. This learned structure is the minimal solution to the XOR problem. The baseline model (c and d) is trained without L_Σ and depicts a less interpretable solution to XOR.

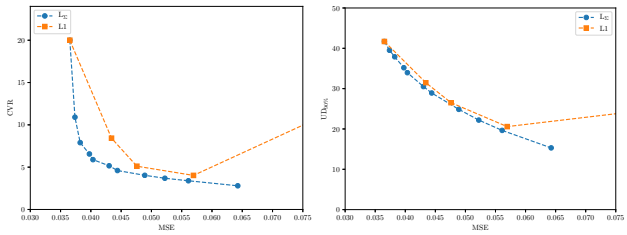


Figure 4: Results on CIFAR-10 test set using L_Σ and L^1, respectively. **Left:** CVR against MSE. **Right:** $UD_{90\%}$ against MSE. Each point in the plot: $\lambda \in [0.0, 0.2, ..., 10.24]$.

	CVR	**UD$_{90\%}$**	**MSE**
L_Σ	6.56	35.18	0.0398
L^1	4.03	20.59	0.0569
W/O	20.00	41.69	0.0365

Table 1: Results from the experiments on CIFAR.

As can be seen in Figure 3 the model was able to learn the optimal structure of exactly 2 dimensions in the first layer and one dimension in the second, whereas the baseline did not. Further, as expected, the first layer do encode a negative covariance between the two active units while the second layer is completely free from covariance. Note that, even though the second hidden layer is not the output of the model it does encode the result in that one active neuron.

3.4 Non-linear uncorrelated convolutional features

Convolutional autoencoders have been used to learn visual features. Here, we will see that it is possible to train a deep convolutional autoencoder on real-world data and learn representations that have low covariance, while retaining the reconstruction quality.

To keep it simple, the encoder part of the model used two convolutional layers and two fully connected layers, with a total of roughly 500.000 parameters in the whole model. The regularization was applied to the coding layer which has 84 dimensions, giving a bottleneck effect. The model was trained and evaluated on the CIFAR-10 dataset (Krizhevsky & Hinton, 2009), containing 32x32 pixel colour images. Official test set: 10,000 images, 5,000 images from the training set of 50,000 was set aside for validation. We compare the results from using L_Σ regularization with L^1 regularization and with no regularization at all (W/O). The model was trained using Adam (Kingma & Ba, 2015) with early stopping. Initial learning rate: 0.001, batch size: 100, λ: 0.08. The reported scores are averages from training the model five times with different initialization.

The results (see Table 1) show that the high-level features become more disentangled and has a lower CVR (6.56) using L_Σ regularization. Without regularization, the score is 20.00, and with L^1 regularization the score is 4.03. The model with L_Σ regularization obtains an MSE of 0.0398, roughly the same as without regularization (0.0365), both of which are much better than using L^1 regularization, with an MSE of 0.0569. Figure 4 shows the CVR score vs the MSE, illustrating that L_Σ leads to more disentangled representations.As you increase the regularization factor L_Σ regularization pushes down the CVR quickly, while retaining an MSE error that is almost constant. L^1 regularization also lower CVR, although slower, and at a higher MSE cost. The $UD_{90\%}$ results show that L_Σ encourages representations that concentrate the variation.

4 Related work

Different notions of independence have been proposed as useful criteria to learn disentangled representations. Principal component analysis (PCA; Pearson, 1901) can find linearly uncorrelated variables. Nonlinear PCA often refers to neural autoencoders (Kramer, 1991) without specific regularization. Independent component analysis (ICA; Hyvärinen et al., 2004) has a somewhat stronger requirement of statistical independence. Dinh et al., (2015; 2017) used the substitution rule of differentiation as a motivation for the model. Using a fixed factorial prior, they encouraged the model to learn independent representations. Brakel & Bengio (2017) used adversarial training to make a generative network learn a factorized, independent distribution $p(\mathbf{z})$. Our approach is more flexible and portable, as it can be applied as a regularization to learn uncorrelated components in any gradient-based model that learns internal representations.

5 Conclusions

In this paper, we have presented L_Σ regularization, a novel regularization scheme based on penalizing the covariance between dimensions of the internal representation learned in a hierarchical model. The proposed regularization scheme helps models learn linearly uncorrelated variables in a non-linear space. While techniques for learning independent components follow criteria that are more strict, our solution is flexible and portable, and can be applied to any feature-learning model that is trained with gradient descent. Our method has no penalty on the performance on tasks evaluated in the experiments, while it does disentangle the data. We saw that our approach performs well applied to a standard deep convolutional autoencoder on the CIFAR-10 dataset (Krizhevsky & Hinton, 2009); the resulting model performs comparable to the model without regularization, while we can also see that the covariances between dimensions in the internal representation decrease drastically.

Acknowledgments

The authors would like to acknowledge the project *Towards a knowledge-based culturomics* supported by a framework grant from the Swedish Research Council (2012–2016; dnr 2012-5738), and the project *Data-driven secure business intelligence* grant IIS11-0089 from the Swedish Foundation for Strategic Research (SSF).

References

A. Achille and S. Soatto. Emergence of Invariance and Disentangling in Deep Representations. *ICML Workshop on Principled Approaches to Deep Learning*, 2017.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, 2013.

Philémon Brakel and Yoshua Bengio. Learning independent features with adversarial nets for non-linear ICA. *arXiv preprint arXiv:1710.05050*, 2017.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear independent components estimation. *ICLR*, 2015.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. *ICLR*, 2017.

Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis*, volume 46. John Wiley & Sons, 2004.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical report, University of Toronto*, 2009.

Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

Greg Ver Steeg and Aram Galstyan. Maximally informative hierarchical representations of high-dimensional data. In *Artificial Intelligence and Statistics*, pp. 1004–1012, 2015.

Learning 6-DOF Grasping Interaction via Deep 3D Geometry-aware Representations

Xinchen Yan[**] **Jasmine Hsu**[‡] **Mohi Khansari**[†] **Yunfei Bai**[†] **Arkanath Pathak**[§]
Abhinav Gupta[×] **James Davidson**[‡] **Honglak Lee**[‡]

Abstract

This paper focuses on the problem of learning 6-DOF grasping with a parallel jaw gripper in simulation. Our key idea is constraining and regularizing grasping interaction learning through 3D geometry prediction. Specifically, we formulate the learning of deep geometry-aware grasping model in two steps: First, we learn to build mental geometry-aware representation by reconstructing the scene (i.e., 3D occupancy grid) from RGBD input via generative 3D shape modeling. Second, we learn to predict grasping outcome with its internal geometry-aware representation. The learned outcome prediction model is used to sequentially propose grasping solutions via analysis-by-synthesis optimization. Our contributions are fourfold: (1) To best of our knowledge, we are presenting for the first time a method to learn a 6-DOF grasping net from RGBD input; (2) We build a grasping dataset from demonstrations in virtual reality with rich sensory and interaction annotations. This dataset includes 101 everyday objects spread across 7 categories, additionally, we propose a data augmentation strategy for effective learning; (3) We demonstrate that the learned geometry-aware representation leads to about 10% relative performance improvement over the baseline CNN on grasping objects from our dataset. (4) We further demonstrate that the model generalizes to novel viewpoints and object instances.

1 Introduction

Learning to interact with and grasp objects is a fundamental and challenging problem in robot learning that combines perception, motion planning, and control. The problem is challenging because it not only requires understanding geometry (the global shape of an object, the local surface around the interaction space) but it also requires estimating physical properties, such as weight, density, and friction. Furthermore, it requires invariance to illumination, object location, and viewpoint. To handle this, current data-driven approaches [8, 15, 10, 13, 12] use hundreds of thousands of examples to learn a solution.

While further scaling may help improve performance of these methods, we postulate shape is core to interaction and that additional shape signals to focus learning will boost performance. The notion of using shape and geometry has been pioneered in grasping research [7, 9, 2, 11, 19].

Inspired by these approaches, we propose the concept of a deep **geometry-aware** representation (e.g., [21, 5, 3, 20, 14, 16, 22, 18, 6, 4]) for grasping. Key to our approach is that we first build a mental representation by *recognizing* and *reconstructing* the 3D geometry of the scene from RGBD input. With the built-in 3D geometry-aware representation, we can hallucinate a local view of the object's geometric surface from the gripper perspective that will be directly useful for grasping interaction. In contrast with black-box models that do not have explicit notion of 3D geometry and prior shape-based grasping approaches, our approach has the following features: (1) it performs

[**]University of Michigan, during internship with Google Brain. [‡]Google Brain, [†]X Inc, [×]Google Research, [§]Google

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

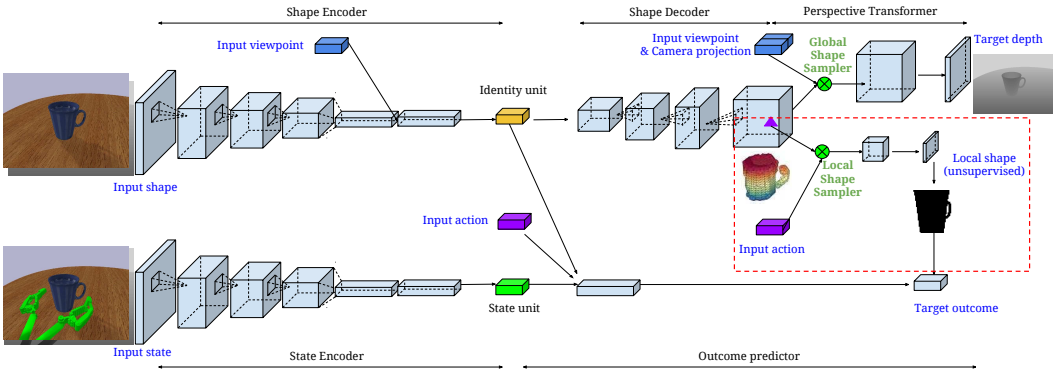


Figure 1: Illustration of deep geometry-aware grasping network.

3D shape reconstruction as an auxiliary task; (2) it hallucinates the local view using a learning-free physical projection operator; and (3) it explicitly reuses the learned geometry-aware representation for grasping outcome prediction.

In this work, we design an end-to-end deep geometry-aware grasping network (see Figure 1) for learning this representation. Our geometry-aware network has two components: a shape generation network and a grasping outcome prediction network. The shape generation network learns to recognize and reconstruct the 3D geometry of the scene with an image encoder and voxel decoder. The image encoder transforms the RGBD input into a high-level geometry representation that involves shape, location, and orientation of the object. The voxel decoder network takes in the geometry representation and outputs the occupancy grid of the object. To further hallucinate the local view from gripper perspective, we propose a novel learning-free image projection layer similar to [22, 16]. Building upon the shape generation network, our grasping outcome prediction network learns to produce a grasping outcome (e.g., success or failure) based on the action (i.e. gripper pose), the current visual state (e.g., object and gripper), and the learned geometry-aware 3D representation. Unlike our end-to-end multi-objective learning framework, existing data-driven grasping pipelines [15, 13, 12] can be viewed as models without a shape generation component. They require either an additional camera to capture the global object shape or extra processing steps, such as object detection and patch alignment. Furthermore, these methods learn over a constrained grasp space, typically either 3-DOF or 4-DOF. We relax this constraint to learn fully generalized 6-DOF grasp poses.

We have built a large database consisting of 101 everyday objects with around 150K grasping demonstrations in Virtual Reality with both human and augmented synthetic interactions. For each object, we collect 10-20 grasping attempts with a parallel jaw gripper from right-handed users. For each attempt, we record a pre-grasping status which includes the location and orientation of the object and gripper, as well as the grasping outcome (e.g., success or failure given if the object is between the gripper fingers after closing and lifting). To acquire sufficient data for learning, we generate additional synthetic data by perturbing the gripper location and orientation from human demonstrations using PyBullet [1]. We plan to open-source the dataset as well as the Tensorflow implementation of our deep geometry-aware grasping network. A demo video of our approach is available at https://goo.gl/gPzPhm.

2 Experimental Setup and Main Results

Implementation details. Our deep grasping network (see Figure 1) is composed of a shape generation network and an outcome prediction network. The shape generation network has a 2D convolutional shape encoder and a 3D deconvolutional shape decoder followed by a global projection layer. The outcome prediction network has a 2D convolutional state encoder and a fully connected outcome predictor with an additional local shape projection layer. We adopt the current data-driven framework as our grasping baseline by removing the shape encoder and shape decoder from our deep geometry-aware grasping model. This baseline can be interpreted as the grasping quality CNN [12] without an additional view from a top-down camera. Both baseline and our geometry-aware model adopt convolutional encoder-decoder architecture with residual connections. The bottleneck layer (e.g., the identity unit in the geometry-aware model) is a 768 dimensional vector. We plan on releasing the Tensorflow implementation of our deep geometry-aware grasping network.

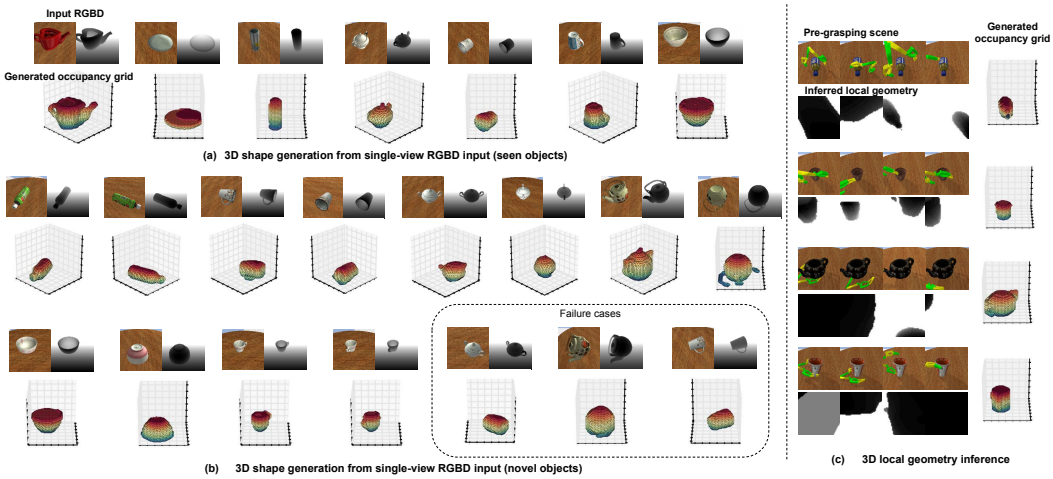


Figure 2: Visualization: 3D shape generation from single-view RGBD. (a) The performance on training (seen) objects. (b) The performance on testing (novel) objects. (c) Local geometry inference from generated occupancy grid.

Method / Category	bottle	bowl	cup	plate	mug	sugarbowl	teapot	all
baseline CNN (15)	72.81	73.36	73.26	66.92	72.23	70.45	66.13	71.42
geo-aware CNN (15)	**78.83**	**79.32**	**77.60**	**68.88**	**78.25**	**76.09**	**73.69**	**76.55**
baseline CNN (45)	71.02	74.16	73.50	63.31	74.23	72.70	64.19	71.32
geo-aware CNN (45)	**78.77**	**80.63**	**78.06**	**70.13**	**79.29**	**77.52**	**72.88**	**77.25**

Table 1: Grasping Outcome prediction accuracy from seen elevation angles.

Method / Category	bottle	bowl	cup	plate	mug	sugarbowl	teapot	all
baseline CNN (30)	71.15	72.98	71.65	61.90	71.01	70.06	61.88	69.50
geo-aware CNN (30)	**79.17**	**77.71**	**77.23**	**67.00**	**75.95**	**75.06**	**70.66**	**75.27**
baseline CNN (60)	68.45	73.05	72.50	61.27	74.40	71.30	63.25	70.18
geo-aware CNN (60)	**77.40**	**78.52**	**76.24**	**68.13**	**79.39**	**76.15**	**70.34**	**75.76**

Table 2: Grasping Outcome prediction accuracy from novel elevation angles.

Visualization: 3D shape generation. We evaluate the quality of the shape generation model by visualizing the geometry representations through the shape encoder and decoder network. In our evaluations, we used single-view RGBD input and corresponding camera view matrix as input to the network. As shown in Figure 2(a), our shape generation model is able to generate a detailed 3D occupancy grid from single-view input without 3D supervision during training. As shown in Figure 2(b), our model demonstrates reasonable generalization quality even on novel object instances.

Analysis: local geometry inference via projection. One advantage of our shape generation component is that we can obtain additional local geometry information (see the red-dashed box in Figure 1(c)) from our geometry-aware representation. This is the key difference between our work and the related work that require additional camera from the gripper. With 3D geometry as part of the intermediate representation, we hallucinate the local geometry by running a projection from the gripper's perspective (i.e., simply treat the gripper as another virtual camera). To further understand the advantages of our shape generation component, we visualized the intermediate local geometry projected from generated 3D occupancy grid. As shown in Figure 2(c), our shape generation component provides accurate local geometry estimation that is useful for grasping outcome prediction.

Model evaluation: Grasping outcome prediction. To evaluate the actual advantages in grasping outcome prediction from our modeling, we computed the average classification accuracy over 30K demonstrations from novel object instances (from testing set) with diverse observation viewpoints. For each human demonstration, we generated 100 synthetic grasps through perturbation (among which 50% of them are success grasps) and computed the average accuracy on 100 grasps (i.e., random guess achieves 50% accuracy). To investigate the model performance due to viewpoint changes, we repeat the evaluation experiment for four different elevation angles (e.g, 15, 30, 45, and 60 degrees). We use parallel computing resources (500 machines) during evaluation and the

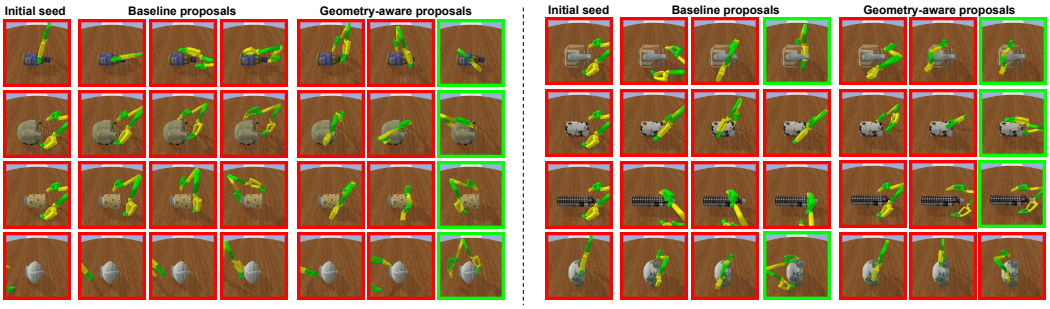


Figure 3: Visualization: grasping optimization with CEM based on the grasping prediction output. In each row, we selected three representative steps in grasping optimization (in sequential order from left to right). Red box represents a failure grasp while green box represents a successful grasp.

Method / Category	bottle	bowl	cup	plate	mug	sugarbowl	teapot	all
baseline CNN + CEM	48.60	64.28	55.44	45.99	61.00	53.97	63.08	55.85
geo-aware CNN + CEM	**56.73**	**68.84**	**60.31**	**50.09**	**67.21**	**59.87**	**69.22**	**61.46**
rel. improvement (%)	16.72	7.09	8.77	8.92	10.18	10.92	9.73	10.03

Table 3: Grasping planning on novel objects: success rate by optimizing for up to 20 steps.

entire evaluation took about 1 day. The results are summarized in Table 1 and Table 2. Overall, the deep geometry-aware model consistently outperforms the deep CNN baseline in grasping outcome classification. As we can see, "teapot" and "plate" are comparatively more challenging categories for outcome prediction, since "teapot" has irregular shape parts (e.g., tip and handle) and "plate" has a fairly flat shape. When it comes to novel elevation angles (e.g., compare Table 1 and Table 2), our deep geometry-aware model is less affected, especially in categories such as "teapot" and "plate" where viewpoint-invariant shape understanding is crucial.

Application: Analysis-by-synthesis grasping planning. As we improve the classification accuracy over the grasping outcome, a natural question is whether this improvement can be used to guide better grasping planning. Given a grasping proposal (defined as target gripper pose) seed, we conducted grasping planning by sequentially adjusting the grasping pose guided by our deep grasping network until a grasp success. In each optimization step, we performed cross-entropy method (CEM) [17, 10] as follows. (1) We initialized with a failure grasp in order to force the model to find better grasping pose. (2) To obtain the gradient direction in the 6D space, we sample 10 random directions and selected the top one based on the score returned by the neural network (output of outcome predictor). We repeat the iterations until success (we set an upper bound of 20 steps). We conducted the same grasping explore evaluation for both the baseline CNN and our deep geometry-aware model. To account for the variations in observation viewpoints and initial seeds, we repeat the evaluation for eight times per testing demonstration in our dataset and reported the average success rate after 20 iterations (marked as failure only if there is no success in 20 steps). As shown in Table 3, CEM guided our geometry-aware model performance consistently better than baseline CNN model. We believe the improved performance comes from the explicit modeling of the 3D geometry as intermediate representation in our deep geometry-aware model. Our model achieved the most significant improvement in the "bottle" category, since a bottle shape is relatively easy to reconstruct. Our improvement in the "bowl" category is less significant, partly due to the difficulty of predicting its concave shape in novel object instances. Figure 3 demonstrates example grasping planning trajectories on different objects. The baseline CNN is less robust compared to our deep geometry-aware model, which is more likely to transit from one side of the object to the other side with a clear notion of 3D geometry.

3 Conclusions

In this work, we studied the problem of learning the grasping interaction with deep geometry-aware representation. We proposed a deep geometry-aware network that performs shape generation as well as grasping outcome prediction with a learning-free physical projection layer. Compared to the CNN baseline, experimental results demonstrated improved performance in outcome prediction thanks to generative shape modeling. Guided by the geometry-aware representation, we obtained better planning via analysis-by-synthesis grasping optimization.

References

[1] Pybullet physics engine. `http://pybullet.org`.

[2] J. Bohg and D. Kragic. Learning grasping points with shape context. *Robotics and Autonomous Systems*, 58(4):362–377, 2010.

[3] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European Conference on Computer Vision*, pages 628–644. Springer, 2016.

[4] M. Gadelha, S. Maji, and R. Wang. 3d shape induction from 2d views of multiple objects. *arXiv preprint arXiv:1612.05872*, 2016.

[5] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016.

[6] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2016.

[7] C. Goldfeder, M. Ciocarlie, H. Dang, and P. K. Allen. The columbia grasp database. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 1710–1716. IEEE, 2009.

[8] I. Lenz, H. Lee, and A. Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015.

[9] B. León, S. Ulbrich, R. Diankov, G. Puche, M. Przybylski, A. Morales, T. Asfour, S. Moisio, J. Bohg, J. Kuffner, et al. Opengrasp: A toolkit for robot grasping simulation.

[10] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, page 0278364917710318.

[11] M. Li, K. Hang, D. Kragic, and A. Billard. Dexterous grasping under shape uncertainty. *Robotics and Autonomous Systems*, 75:352–364, 2016.

[12] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arxiv preprint: 1703.09312*, 2017.

[13] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1957–1964. IEEE, 2016.

[14] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 922–928. IEEE, 2015.

[15] L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 3406–3413. IEEE, 2016.

[16] D. J. Rezende, S. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess. Unsupervised learning of 3d structure from images. In *Advances In Neural Information Processing Systems*, pages 4997–5005, 2016.

[17] R. Rubinstein and D. Kroese. The cross-entropy method: A unified approach to combinatorial optimization, monte-carlo simulation, and machine learning. 2004.

[18] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *CVPR*, 2017.

[19] N. Vahrenkamp, L. Westkamp, N. Yamanobe, E. E. Aksoy, and T. Asfour. Part-based grasp planning for familiar objects. In *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*, pages 919–925. IEEE, 2016.

[20] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016.

[21] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.

[22] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Advances in Neural Information Processing Systems*, pages 1696–1704, 2016.

Disentangling by Factorising

Hyunjik Kim[1,2], **Andriy Mnih**[1]
DeepMind[1], University of Oxford[2]
hyunjikk@google.com, amnih@google.com

Abstract

We introduce FactorVAE, a method that disentangles by encouraging the distribution of codes, the variational posterior averaged over the training set, to be factorial and hence independent in the dimensions. Furthermore, we propose a new measure of disentanglement that addresses some weaknesses of commonly used metrics.

1 Introduction

Learning interpretable representations of data that expose semantic meaning has important consequences for artificial intelligence. Such representations are useful not only for standard downstream tasks such as supervised learning and reinforcement learning, but also for tasks such as transfer learning and zero-shot inference where humans excel but machines struggle [14]. In particular, there have been multiple efforts in the deep learning community towards learning factors of variation in the data, commonly referred to as learning a *disentangled representation*. While there is no canonical definition for this term, we adopt the following definition: a representation where a change in one dimension corresponds to a change in one factor of variation, while being relatively invariant to change in other factors [4]. Moreover, we focus on image data in this work.

Using generative models has shown great promise in learning disentangled representations in images. Notably semi-supervised approaches that require implicit or explicit knowledge about the true underlying factors of the data have excelled at disentangling [13, 11, 23, 24]. However, ideally we would like to learn these in an unsupervised manner, due to the following reasons: 1. Humans are able to learn factors of variation unsupervised [22]. 2. Labels are costly as obtaining them requires a human in the loop. 3. Labels assigned by humans may be inconsistent and could also lead to omissions of factors that are imperceptible to the human eye.

The generative models used for unsupervised disentangling largely fall into two categories: the Variational Autoencoder (VAE) framework [12] and the Generative Adversarial Net (GAN) framework [8]. InfoGAN [6] is a notable example of the latter that learns disentangled representations by encouraging mutual information between the observations and a subset of latent variables. However it suffers from instabilities in training, and its disentangling performance is sensitive to the choice of the prior and the number of latents used [9]. The β-VAE [9] uses a VAE objective with extra penalty on the KL between the variational posterior and the prior, giving a more robust and stable method of disentangling.

One drawback of the β-VAE is that there is a strong dependency between disentanglement and reconstruction. The motivation for our work is to get a better trade-off between disentanglement and reconstruction, so as to improve the optimal disentanglement and also obtain sharper reconstructions. We achieve this goal by first analysing the source of this trade-off. We then propose FactorVAE, which modifies the objective accordingly, introducing a penalty that encourages the marginal distribution of representations to be factorial without hurting reconstruction too much. This new penalty is optimised using a discriminator network, following the divergence minimisation view of GANs [19, 21], and we show that our approach enhances disentanglement as well as reconstruction compared to the β-VAE. Moreover, to help quantify our improvements, we point out the weaknesses in existing metrics of disentanglement, and propose a new metric that addresses these shortcomings.

2 The Trade-off between Disentanglement and Reconstruction in β-VAE

We motivate our approach by analysing where the disentanglement and reconstruction trade-off arises in the β-VAE loss. First, we introduce notation and architecture of our VAE framework. We have observations $x^{(i)}, i = 1, \dots, N$ in image space $\mathcal{X}$, and latents $z \in \mathbb{R}^D$ are real vectors interpreted as representations of the data. The generative model is defined by the standard Gaussian prior $p(z) = \mathcal{N}(0, I)$, and the decoder $p_\theta(x|z)$ parameterised by a DeconvNet with weights θ. The variational posterior is given by the encoder $q_\phi(z|x) = \prod_{j=1}^{D} \mathcal{N}(z_j|\mu_j(x), \sigma_j^2(x))$, parameterised by a ConvNet with weights ϕ. An important distribution for our analysis is the marginal posterior of VAEs, namely the marginal distribution of the latents/code (used interchangeably):

$$r(z) = \int p_{data}(x) q(z|x) dx = \frac{1}{N} \sum_{i=1}^{N} q(z|x^{(i)}) \tag{1}$$

We can easily sample from r by ancestral sampling: $x \sim p_{data}, z \sim q(\cdot|x)$. r is relevant for when we are looking for a disentangled representation; should the representations correspond to the independent factors of variation in the data, we would like the distribution of these representations to be factorised, i.e. independent in the dimensions: $r(z) = \prod_{j=1}^{D} r(z_j)$.

The β-VAE objective is as follows:

$$\sum_{i=1}^{N} \mathbb{E}_{q(z|x^{(i)})}[\log p(x^{(i)}|z)] - \beta KL(q(z|x^{(i)})||p(z)) \tag{2}$$

Note this is a variational lower bound for $\beta \geq 1$. The first term is a measure of reconstruction, and the second term is the complexity penalty that acts as a regulariser. We may break down this KL term further as follows [10, 15]:

$$KL(q(z|x^{(i)})||p(z)) = I(x; z) + KL(r(z)||p(z)) \tag{3}$$

where $I(x; z)$ is the mutual information (MI) between x and z under the joint distribution of the data and their codes $p_{data}(x)q(z|x)$. The $KL(r(z)||p(z))$ term encourages independence in the dimensions of z and hence disentanglement by pushing $r(z)$ towards $p(z)$, a factorised distribution. On the other hand, penalising the MI term $I(x; z)$ acts as an information bottleneck between x and z whose presence is necessary for generalisation, but penalising this term too heavily (high β) leads to a lack of information about x in z and hence poor reconstruction [15]. Hence initially raising β from 1, and thus further penalising both terms, leads to better disentanglement while sacrificing reconstruction. When this sacrifice is severe, there is insufficient information about the observation in the latents, hurting disentanglement as well. So there exists an optimal value of $\beta > 1$ that gives highest disentanglement, whose reconstructions are blurrier than a VAE ($\beta = 1$).

3 The Total Correlation penalty and FactorVAE

We motivate FactorVAE with the suspicion that the further penalty on the MI might be unnecessary for improved disentanglement. So instead, we keep the VAE objective and directly encourage independence in the code distribution, arriving at our new objective:

$$\sum_{i=1}^{N} \mathbb{E}_{q(z|x^{(i)})}[\log p(x^{(i)}|z)] - KL(q(z|x^{(i)})||p(z)) - \gamma KL(r(z)||\prod_{j=1}^{D} r(z_j)) \tag{4}$$

The latter term is also known as *Total Correlation* (TC) [27], often used as a measure of independence for multiple random variables. However this term is intractable, so we need further machinery to optimise it. To do so, first note that we can easily sample from $r(z)$ using ancestral sampling described above. Moreover we can sample from $\prod_j r(z_j)$ by sampling D times from $r(z)$ then ignoring all but one dimension for each sample, or more efficiently by sampling a batch from $r(z)$ then randomly permuting across the batches for each dimension. As long as the batch is large enough, these samples will be close to sampling from $\prod_j r(z_j)$. This is a standard trick used in the independence testing literature [2, 7]. Having access to samples from both distributions allows us to minimise their KL

divergence using a discriminator to approximate the density ratio that arises in the KL [20, 26]. That is to say, suppose we have a discriminator D_ψ, an MLP with weights ψ, that outputs a probability given input z. Suppose it approximates the probability that z is a sample from $r(z)$ over $\prod_j r(z_j)$. Then we have:

$$TC(z) = KL(r(z)||\prod_{j=1}^{D} r(z_j)) = \mathbb{E}_{r(z)}\left[\log \frac{r(z)}{\prod_j r(z_j)}\right] \approx \mathbb{E}_{r(z)}\left[\log \frac{D(z)}{1-D(z)}\right] \tag{5}$$

So for FactorVAE we train the discriminator and the VAE by simultaneous gradient descent. In particular, the VAE parameters θ, ϕ are updated using the loss in Equation 4, but replacing the TC term by the right hand side of Equation 5. The discriminator is trained using samples from $r(z)$ and $\prod_j r(z_j)$ to approximate their density ratio and hence the TC.

Note that in the usual GAN literature, the divergence minimisation occurs between two distributions over the data space, which is often very high dimensional (e.g. images). So the two distributions often have disjoint support, which makes training unstable especially when the discriminator is strong. Hence it is necessary to use tricks such as using sparse discriminator updates, instance noise [25] or getting rid of the discriminator altogether as for Wasserstein-divergence [3]. For our work, we are minimising divergence between two distributions over the latent space (as in e.g. [18]), which is usually much lower dimensional and the two distributions have overlapping support. We observe that training is stable for large enough batch sizes, allowing us to use a strong discriminator with frequent updates.

4 A New metric for Disentanglement

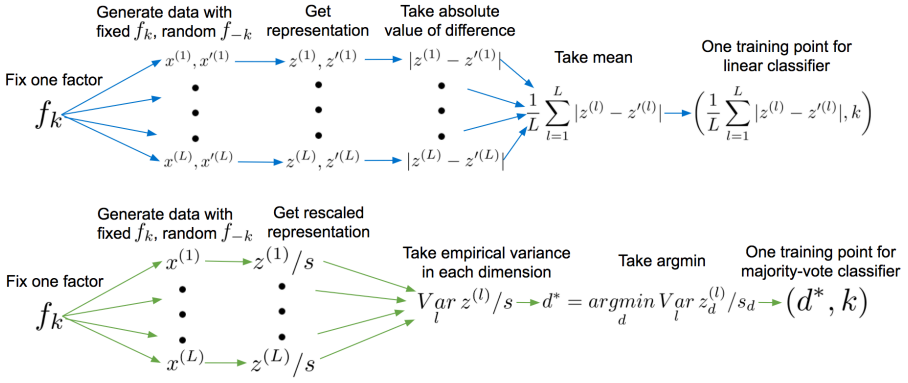


Figure 1: Top: Metric in [9]. Bottom: Our new metric, where $s \in \mathbb{R}^d$ is the scale (empirical standard deviation) of latent representations of the full data (or big enough random subset)

Returning to our definition of disentanglement, where a change in one dimension of the representation corresponds to a change in precisely one factor of variation, we point out that this definition is quite crude. We have implicitly ignored the possibility of: correlations among the factors, hierarchy in the factors of variation, and a many-to-one mapping between a combination of factors and a data point (over-representation). Thus our definition is limited to synthetic data with independent factors of variation. However, as we'll show in the paper, robust disentanglement is not a fully solved problem even in this setting. Part of the obstacle in achieving this first milestone lies in the absence of a sound, quantitative metric for measuring disentanglement. We point out the weaknesses in existing methods of assessing disentanglement, and introduce a new metric that addresses these problems.

A popular method of measuring disentanglement is by inspecting latent traversals: visualising the change in reconstructions as one traverses across each dimension of the latent space. The qualitative nature of this approach makes it unsuitable for comparing different algorithms. Moreover we must look at multiple latent traversals for a robust assessment of an algorithm, namely using multiple reference images, random seeds, and points during training. Having a human in the loop to assess the traversals is too time consuming and the evaluations are not reproducible.

The authors of β-VAE proposed a supervised metric that attempts to quantify disentanglement [9]. The metric is the error rate of a linear classifier that is trained as follows. Choose a factor k; generate

data with this factor fixed but all other factors varying randomly; obtain their representations; take the absolute value of the pairwise differences of these representations. Then the mean of these statistics across the pairs gives one training input for the classifier, and the fixed factor k is the corresponding training output (see top of Figure 1). So if the representations were perfectly disentangled, we would see zeroes in the dimension of the training input that corresponds to the fixed factor of variation, and the linear classifier will learn to map the index of the zero value to the factor.

However this metric has several weaknesses. Firstly, the metric could be sensitive to hyperparameters of the linear classifier optimisation (optimiser, weight initialisation, number of training iterations) hence these need to be tuned. Secondly, having a linear classifier is not so intuitive - we could get representations where each factor corresponds to a linear combination of dimensions instead of a single dimension. Finally and most importantly, the metric has a failure mode where it gives 100% accuracy when it only disentangles $K - 1$ factors out of K; for the remaining factor, the classifier can cheat by detecting when all dimensions for the $K - 1$ factors are non-zero. An example of such a case is displayed in Figure 2.

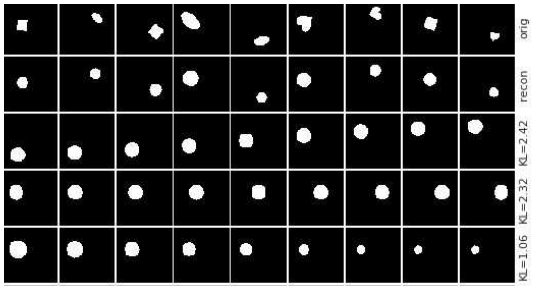


Figure 2: A model trained on the 2D Shapes data that scores 100% on metric in [9] (ignoring the shape factor). First row: originals. Second row: reconstructions. Remaining rows: reconstructions of latent traversals. The model captures x-pos,y-pos and scale but ignores orientation, yet achieves a perfect score on the metric.

So we propose an enhanced disentanglement metric as follows. Choose a factor k; generate data with this factor fixed but all other factors varying randomly; obtain their representations; rescale each dimension by its empirical standard deviation of representations over the full data (or a large enough random subset); take the empirical variance in each dimension. Then the index of the dimension with lowest variance gives one training input with training output k for a classifier. So if the representation is perfectly disentangled, the empirical variance in the dimension corresponding to the fixed factor will be 0. We rescale the representations prior to taking the argmin, so that the argmin is invariant to rescaling of the representations in each dimension. Since both inputs and outputs lie on a discrete space, the optimal classifier is the majority-vote classifier, and the metric is the error rate of the classifier. Here the classifier is a deterministic function of the training data, hence there are no optimisation hyperparameters to tune, and we claim that the metric is conceptually simpler and more intuitive than the previous metric. Most importantly it circumvents the failure mode in the latter, since the classifier needs to see the lowest variance in a latent dimension for a given factor to classify it correctly (see bottom of Figure 1).

5 Related Work

Using a discriminator to optimise a divergence encouraging independendence has been explored in a couple of recent works. The Adversarial Autoencoder (AAE) [16] removes the MI term in the VAE objective, to optimise the reconstruction error plus $KL(r(z)||p(z))$ using the density ratio trick. However they explore the AAE for semi-supervised classification or unsupervised clustering, not so much in the context of disentangling. In PixelGAN Autoencoders [15] that use the same objective, it is claimed that adding extra additive noise to the inputs of the encoder is crucial, which could be an indication that an information bottleneck is necessary and that the MI term shouldn't be dropped. Brakel et al [5] also use a discriminator to minimise the Jensen-Shannon Divergence between the distribution of codes and the product of its marginals, but use the GAN framework with deterministic encoders and decoders, and only explore their technique in the context of Independent Component

Analysis source separation. Achille et al [1] use a similar objective to FactorVAE but with β in front of the $KL(q(z|x)||p(z))$ and sets $\beta = \gamma$ for tractability.

6 Preliminary Experiments

We show results of experiments on the 2D Shapes dataset [17], which are binary 64 by 64 images generated from five factors of variation: shape, x-position, y-position, scale and orientation. The encoder is a 4-layer ConvNet and the decoder is a deConvNet with the same architecture (same as in [9]). The discriminator is a 6-layer MLP with 1000 units per layer, and use five discriminator updates per VAE update (smaller MLPs and fewer discriminator updates work fine, but we noticed slight improvements up to this setting).

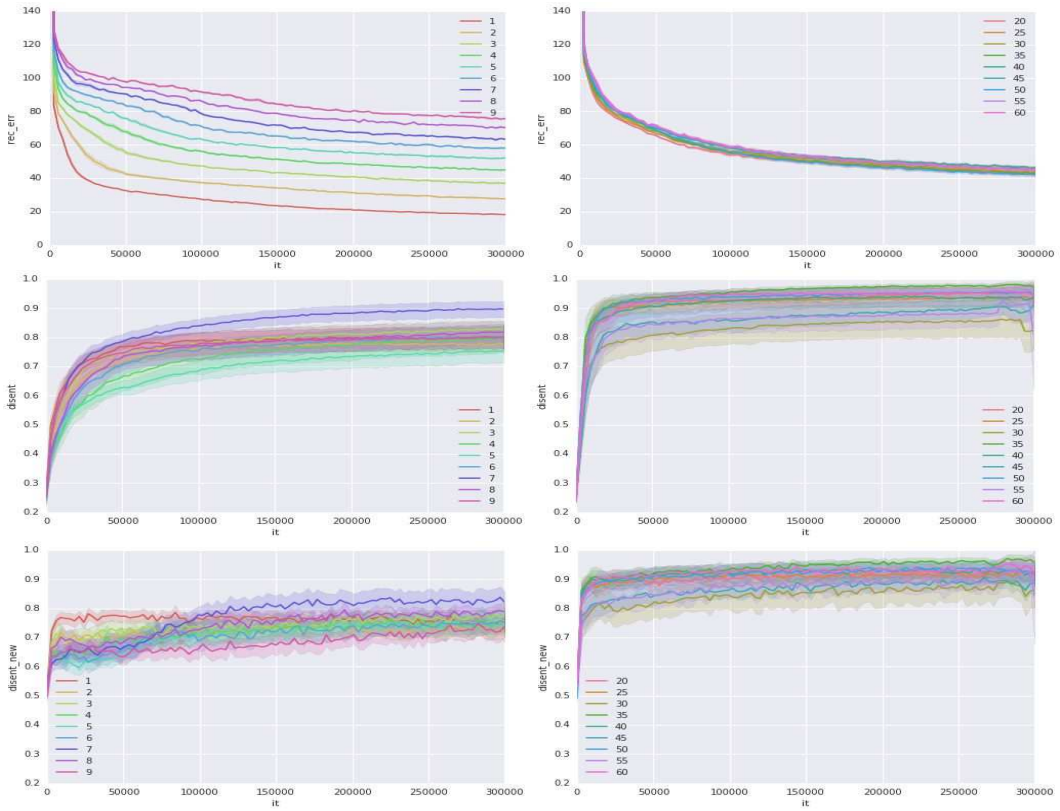


Figure 3: Reconstruction error (top), metric in [9] (middle), our metric (bottom). β-VAE (left), FactorVAE (right). The colours correspond to different values of β and γ respectively, and confidence intervals are over 10 random seeds.

From Figure 3, we see that FactorVAE gives much improved disentanglement compared to β-VAE for both metrics[1], and that we can do so without sacrificing reconstruction error too much. Note that the reconstruction error for the best disentanglement of β-VAE ($\beta = 7$) is over 60, which is significantly higher than that for FactorVAE ($\gamma = 35$), around 40. Also comparing the β-VAE scores on the two metrics, we can see that the metric in [9] is inflated compared to our new metric, due to the failure mode described in Section 4 (can be seen from visual inspection). Our method, on the contrary, robustly disentangles the four continuous factors, hence the two metrics are similar.

[1]Both metrics ignore the shape factor for now, since neither the β-VAE nor our method can successfully model discrete factors of variation. This would require using discrete latent variables instead of Gaussians, but jointly modelling discrete and continuous factors of variation is a non-trivial problem that needs further research.

7 Future Work

To ensure that the discriminator is giving us the correct gradients, we wish to investigate the error in the discriminator's approximation of TC and analyse how it evolves during training, and how it is affected by the architecture of the discriminator and the frequency of discriminator updates. We will also carry out experiments for more complex data sets, and provide comparisons with InfoGAN.

References

[1] Alessandro Achille and Stefano Soatto. Information dropout: learning optimal representations through noise. *arXiv preprint arXiv:1611.01353*, 2016.

[2] Miguel A Arcones and Evarist Gine. On the bootstrap of u and v statistics. *The Annals of Statistics*, pages 655–674, 1992.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, pages 214–223, 2017.

[4] 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.

[5] Philemon Brakel and Yoshua Bengio. Learning independent features with adversarial nets for non-linear ica. *arXiv preprint arXiv:1710.05050*, 2017.

[6] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*, pages 2172–2180, 2016.

[7] Gary Doran, Krikamol Muandet, Kun Zhang, and Bernhard Schölkopf. A permutation-based kernel conditional independence test. In *UAI*, pages 132–141, 2014.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014.

[9] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

[10] Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NIPS*, 2016.

[11] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *NIPS*, 2014.

[12] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. 2014.

[13] Tejas D Kulkarni, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In *NIPS*, 2015.

[14] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, pages 1–101, 2016.

[15] Alireza Makhzani and Brendan Frey. Pixelgan autoencoders. 2017.

[16] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[17] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

[18] L. Mescheder, S. Nowozin, and A. Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In *ICML*, 2017.

[19] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[20] XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 2010.

[21] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *NIPS*, pages 271–279, 2016.

[22] G Perry, ET Rolls, and SM Stringer. Continuous transformation learning of translation invariant representations. *Experimental brain research*, 204(2):255–270, 2010.

[23] Scott Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. Learning to disentangle factors of variation with manifold interaction. In *ICML*, pages 1431–1439, 2014.

[24] N Siddharth, Brooks Paige, Van de Meent, Alban Desmaison, Frank Wood, Noah D Goodman, Pushmeet Kohli, and Philip HS Torr. Learning disentangled representations with semi-supervised deep generative

models. In *NIPS*, 2017.

[25] Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised map inference for image super-resolution. In *ICLR*, 2016.

[26] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. Density-ratio matching under the bregman divergence: a unified framework of density-ratio estimation. *Annals of the Institute of Statistical Mathematics*, 64(5):1009–1044, 2012.

[27] Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of research and development*, 4(1):66–82, 1960.

Semantically Decomposing the Latent Spaces of Generative Adversarial Networks

Chris Donahue
Department of Music
University of California, San Diego
cdonahue@ucsd.edu

Zachary C. Lipton
Amazon AI
Carnegie Mellon University
zlipton@cmu.edu

Akshay Balsubramani
Department of Genetics
Stanford University
abalsubr@stanford.edu

Julian McAuley
Department of Computer Science
University of California San Diego
jmcauley@eng.ucsd.edu

Abstract

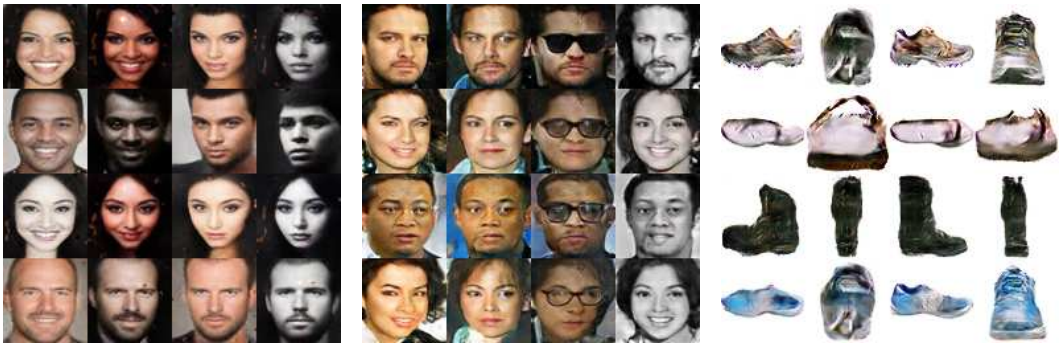
We propose a new algorithm for training generative adversarial networks that jointly learns latent codes for both identities (e.g. individual humans) and observations (e.g. specific photographs). By fixing the identity portion of the latent codes, we can generate diverse images of the same subject, and by fixing the observation portion, we can traverse the manifold of subjects while maintaining contingent aspects such as lighting and pose. Our algorithm features a pairwise training scheme in which each sample from the generator consists of two images with a common identity code. Corresponding samples from the real dataset consist of two distinct photographs of the same subject. In order to fool the discriminator, the generator must produce pairs that are photorealistic, distinct, and appear to depict the same individual. We augment both the DCGAN and BEGAN approaches with Siamese discriminators to facilitate pairwise training. Experiments with an off-the-shelf face verification system demonstrate our algorithm's ability to generate convincing, identity-matched photographs.

1 Introduction

In many domains, a suitable generative process might consist of several stages. To generate a photograph of a product, we might wish to first sample from the space of products, and then from the space of photographs *of that product*. Given such disentangled representations in a multistage generative process, an online retailer might diversify its catalog, depicting products in a wider variety of settings. A retailer could also flip the process, imagining new products in a fixed setting. Datasets for such domains often contain many labeled *identities* with fewer *observations* of each (e.g. a collection of face portraits with thousands of people and ten photos of each). While we may know the identity of the subject in each photograph, we may not know the *contingent* aspects of the observation (such as lighting, pose and background). This kind of data is ubiquitous; given a set of commonalities, we might want to incorporate this structure into our latent representations.

Generative adversarial networks (GANs) learn mappings from latent codes $\mathbf{z}$ in some low-dimensional space $\mathcal{Z}$ to points in the space of natural data $\mathcal{X}$ [6]. They achieve this power through an adversarial training scheme pitting a generative model $G : \mathcal{Z} \mapsto \mathcal{X}$ against a discriminative model $D : \mathcal{X} \mapsto [0, 1]$ in a minimax game. Since their introduction, GANs have been used to generate increasingly high-quality images [13, 17, 2]. However, in their original form, they do not explicitly disentangle the latent factors according to known commonalities. Conditional GANs [11, 12] learn such disentangled

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.



(a) SD-BEGAN trained on faces (b) SD-DCGAN trained on faces (c) SD-DCGAN trained on shoes

Figure 1: Generated samples from SD-GANs. Each matrix represents four distinct identity codes $\mathbf{z}_I$ (one per row) and four distinct observation codes $\mathbf{z}_O$ (one per column).

representations, but are limited to generating instances of existing classes in the data. Several methods propose GANs for learning disentangled representation of faces [15, 8, 16, 1], but all *explicitly* disentangle known contingent factors such as rotation, lighting and age. See [5] for a complete description of related work.

In this paper, we propose Semantically Decomposed GANs (SD-GANs), which encourage a specified portion of the latent space to correspond to a known source of variation.[1,2] Our technique decomposes the latent code $\mathcal{Z}$ into one portion $\mathcal{Z}_I$ corresponding to identity, and the remaining portion $\mathcal{Z}_O$ corresponding to the other contingent aspects of observations. SD-GANs learn through a pairwise training scheme in which each sample from the real dataset consists of two distinct images with a common identity. Each sample from the generator consists of a pair of images with common $\mathbf{z}_I \in \mathcal{Z}_I$ but differing $\mathbf{z}_O \in \mathcal{Z}_O$. In order to fool the discriminator, the generator must not only produce diverse and realistic images, but also images that depict the same identity when $\mathbf{z}_I$ is fixed. For SD-GANs, we modify the discriminator so that it can determine whether a pair of samples constitutes a match.

As a case study, we experiment with a dataset of face photographs, demonstrating that SD-GANs can generate contrasting images of the same subject (Figures 1a, 1b). The generator learns that certain properties are free to vary across observations but not identity. For example, SD-GANs learn that pose, facial expression, hair styles, grayscale vs. color, and lighting can all vary across different photographs of the same individual. We demonstrate that SD-GANs trained on faces generate stylistically-contrasting, identity-matched image pairs that a state-of-the-art face verification algorithm recognizes as depicting the same subject. We also train SD-GANs on a dataset of product images, containing multiple photographs of each product from various perspectives (Figure 1c).

2 Semantically Decomposed Generative Adversarial Networks

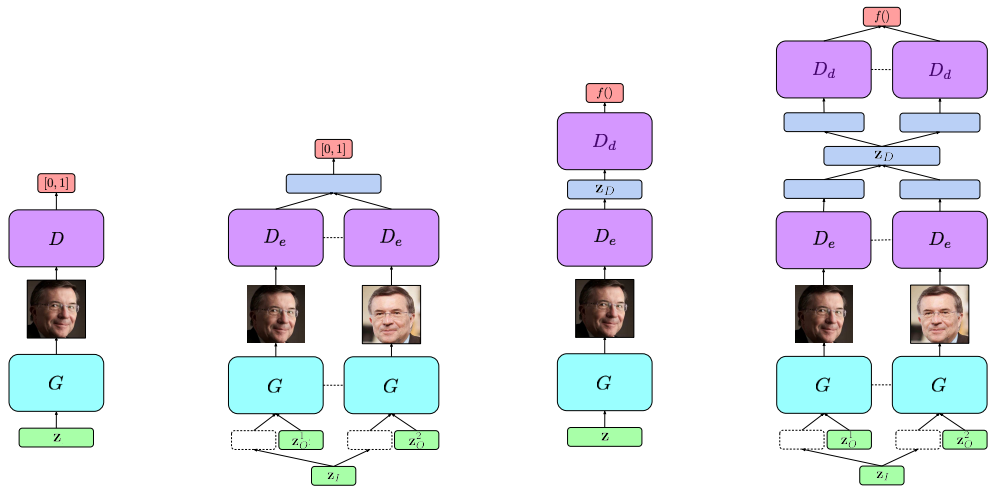
GANs leverage the discriminative power of neural networks to learn generative models. The generative model G ingests latent codes $\mathbf{z}$, sampled from some known prior $P_{\mathcal{Z}}$, and produces $G(\mathbf{z})$, a sample of an implicit distribution P_G. The learning process consists of a minimax game between G, parameterized by θ_G, and a discriminative model D, parameterized by θ_D. In the original formulation, the discriminative model tries to maximize log likelihood, yielding

$$\min_G \max_D V(G, D) = \mathbb{E}_{\mathbf{x} \sim P_R}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim P_{\mathcal{Z}}}[\log(1 - D(G(\mathbf{z})))]. \tag{1}$$

Training proceeds as follows: For k iterations, sample one minibatch from the real distribution P_R and one from the distribution of generated images P_G, updating discriminator weights θ_D to increase $V(G, D)$ by stochastic gradient ascent. Then sample a minibatch from $P_{\mathcal{Z}}$, updating θ_G to decrease $V(G, D)$ by stochastic gradient descent.

[1] Interactive web demo: `https://chrisdonahue.github.io/sdgan`

[2] Source code: `https://github.com/chrisdonahue/sdgan`



(a) DCGAN (b) SD-DCGAN (c) BEGAN (d) SD-BEGAN

Figure 2: SD-GAN architectures and vanilla counterparts. Our SD-GAN models incorporate a decomposed latent space and Siamese discriminators. Dashed lines indicate shared weights. Discriminators also observe real samples in addition to those from the generator (not pictured for simplicity).

2.1 SD-GAN formulation

Consider the data's identity as a random variable I in a discrete index set $\mathcal{I}$. We seek to learn a latent representation that conveniently decomposes the variation in the real data into two parts: 1) due to I, and 2) due to the other factors of variation in the data, packaged as a random variable O. Ideally, the decomposition of the variation in the data into I and O should correspond exactly to a decomposition of the latent space $\mathcal{Z} = \mathcal{Z}_I \times \mathcal{Z}_O$. This would permit convenient interpolation and other operations on the inferred subspaces $\mathcal{Z}_I$ and $\mathcal{Z}_O$.

Our SD-GAN method learns such a latent space decomposition, partitioning the coordinates of $\mathcal{Z}$ into two parts representing the subspaces, so that any $\mathbf{z} \in \mathcal{Z}$ can be written as the concatenation $[\mathbf{z}_I; \mathbf{z}_O]$ of its identity representation $\mathbf{z}_I \in \mathbb{R}^{d_I} = \mathcal{Z}_I$ and its contingent aspect representation $\mathbf{z}_O \in \mathbb{R}^{d_O} = \mathcal{Z}_O$. SD-GANs achieve this through a pairwise training scheme in which each sample from the real data consists of $\mathbf{x}_i^1, \mathbf{x}_i^2 \sim P_R(\mathbf{x} \mid I = i)$, a pair of images with a common identity $i \in \mathcal{I}$. Each sample from the generator consists of $G(\mathbf{z}_I^1), G(\mathbf{z}_I^2) \sim P_G(\mathbf{z} \mid \mathcal{Z}_I = \mathbf{z}_I)$, a pair of images generated from a common identity vector $\mathbf{z}_I \in \mathcal{Z}_I$ but i.i.d. observation vectors $\mathbf{z}_O^1, \mathbf{z}_O^2 \in \mathcal{Z}_O$. We assign identity-matched pairs from P_R the label 1 and $\mathbf{z}_I$-matched pairs from P_G the label 0, yielding

$$V(G, D) = \mathbb{E}_{\mathbf{x}_i^1, \mathbf{x}_i^2 \sim P_R}[\log D(\mathbf{x}_i^1, \mathbf{x}_i^2)] + \mathbb{E}_{\mathbf{z}_I, \mathbf{z}_O^1, \mathbf{z}_O^2 \sim P_{\mathcal{Z}}}[\log(1 - D(G(\mathbf{z}_I; \mathbf{z}_O^1), G(\mathbf{z}_I; \mathbf{z}_O^2)))]. \quad (2)$$

See Algorithm 1 from [5] for SD-GAN training pseudocode.

2.2 SD-GAN discriminator architecture

With SD-GANs, there is no need to alter the architecture of the generator. However, the discriminator must now act upon two images, producing a single output. Moreover, the effects of the two input images $\mathbf{x}_i^1, \mathbf{x}_i^2$ on the output score are not independent. Two images might be otherwise photorealistic but deserve rejection because they clearly depict different identities. To this end, we devise two novel discriminator architectures to adapt DCGAN [13] and BEGAN [2] respectively. In both cases, we first separately encode each image using the same convolutional neural network D_e (Figure 2). We choose this Siamese setup [3, 4] as our problem is symmetrical in the images.

To adapt DCGAN, we stack the feature maps $D_e(\mathbf{x}_i^1)$ and $D_e(\mathbf{x}_i^2)$ along the channel axis, applying one additional strided convolution. This allows the network to further aggregate information from the two images before flattening and fully connecting to a sigmoid output. To adapt BEGAN, we concatenate the encoder representations $[D_e(\mathbf{x}_i^1); D_e(\mathbf{x}_i^2)] \in \mathbb{R}^{2(d_I + d_O)}$ and apply one fully connected bottleneck layer $\mathbb{R}^{2(d_I+d_O)} \Rightarrow \mathbb{R}^{d_I+2d_O}$ with linear activation. Following the bottleneck,

Dataset	AUC	Acc.
MS-Celeb-1M	.913	.867
SD-DCGAN	.823	.749
SD-BEGAN	.928	.857

Table 1: Face verification results for 10k pairs from MS-Celeb-1M (real) and SD-GANs (generated)

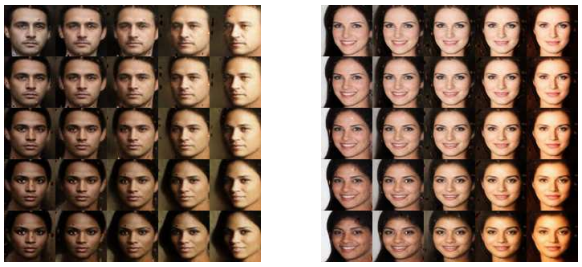


Figure 3: Linear interpolation matrices of identity ($\mathbf{z}_I$ fixed per row) and observation ($\mathbf{z}_O$ fixed per column) for two pairs using SD-BEGAN generator.

we apply a second FC layer $\mathbb{R}^{d_I+2d_O} \Rightarrow \mathbb{R}^{2(d_I+d_O)}$, taking the first $d_I + d_O$ components of its output to be the input to the first decoder and the second $d_I + d_O$ components to be the input to the second decoder. This shared intermediate layer gives SD-BEGAN a mechanism to push apart matched and unmatched pairs.

3 Experiments

We experimentally validate SD-GANs using two datasets: 1) the *MS-Celeb-1M* dataset of celebrity face images [7] and 2) a dataset of shoe images collected from Amazon [10]. From the aligned face images in the MS-Celeb-1M dataset, we select 12,500 celebrities at random and 8 associated images of each, resizing them to 64x64 pixels. We split the celebrities into subsets of 10,000 (training), 1,250 (validation) and 1,250 (test). From the Amazon data, we choose to study shoes as a prototypical example of a category of products. There are around 3000 shoes with multiple product images (6.2 photos of each on average); we use all of them for training. For both datasets, we scale the pixel values to $[-1, 1]$, performing no additional preprocessing or data augmentation.

We train SD-DCGANs on both of our datasets for 500,000 iterations using batches of 16 identity-matched pairs. To optimize SD-DCGAN, we use the Adam optimizer [9] with hyperparameters $\alpha = 2\text{e}{-4}, \beta_1 = 0.5, \beta_2 = 0.999$ as recommended by Radford et al. [13]. We also train an SD-BEGAN on both of our datasets, using the Adam optimizer with the default hyperparameters from [9]. For SD-GANs, we partition the latent codes according to $\mathbf{z}_I \in \mathbb{R}^{d_I}, \mathbf{z}_O \in \mathbb{R}^{d_O}$ using $d_I = d_O = 50$. As in [13], we sample latent vectors $\mathbf{z} \sim \text{Uniform}([-1, 1]^{100})$. For full descriptions of our SD-GAN model architectures, see [5].

To evaluate our results, we procure *FaceNet*, a publicly-available face verifier based on [14].[3] The training objective for FaceNet is to learn embeddings that minimize the L_2 distance between matched pairs of faces and maximize the distance for mismatched pairs. FaceNet ingests images and produces a feature vector $f(\mathbf{x}) \in \mathbb{R}^{128}$. Given two images, $\mathbf{x}_1$ and $\mathbf{x}_2$, we label them as a match if $||f(\mathbf{x}_1) - f(\mathbf{x}_2)||_2^2 \leq \tau_v$, where τ_v is the accuracy-maximizing threshold on a class-balanced set of pairs from MS-Celeb-1M validation data. From MS-Celeb-1M test data and our trained SD-DCGAN and SD-BEGAN models, we evaluate sets of 10k class-balanced pairs (5k matched, 5k unmatched), by reporting the AUC and accuracy (at threshold τ_v) of FaceNet (Table 1). Our results demonstrate that FaceNet verifies generated data from SD-DCGAN and SD-BEGAN similarly to real data; statistics for SD-BEGAN are closer to real data than those of SD-DCGAN. For a full description of our experiments and results of evaluation by human judges, see [5].

4 Discussion

Our results demonstrate that SD-GANs can disentangle those factors of variation corresponding to identity from the rest. In Figure 1b, we demonstrate that by varying the observation vector $\mathbf{z}_O$, SD-GANs can change the color of clothing, add or remove sunglasses, or change facial pose. They can also perturb the lighting, color saturation, and contrast of an image, all while keeping the apparent identity fixed. Moreover, with SD-GANs we can sample never-before-seen identities, a benefit not

[3] "20170214-092102" pretrained model from `https://github.com/davidsandberg/facenet`

shared by conditional GANs. We note, subjectively, that samples from SD-DCGAN (Figure 1b) tend to appear less photorealistic than those from SD-BEGAN (Figure 1a). Given a generator trained with SD-GAN, we can independently interpolate along the identity and observation manifolds (Figure 3). Here, the diagonal represents the entangled interpolation typically shown for ordinary GANs.

On the shoe dataset, we find that the SD-DCGAN model produces convincing results. Manipulating $\mathbf{z}_I$ while keeping $\mathbf{z}_O$ fixed yields distinct shoes in consistent poses (Figure 1c). The identity code $\mathbf{z}_I$ appears to capture the broad categories of shoes (sneakers, flip-flops, boots, etc.). Surprisingly, neither original BEGAN nor SD-BEGAN can produce diverse shoe images.

In this paper, we presented SD-GANs, a new algorithm capable of disentangling factors of variation according to known commonalities. We see several promising directions for future work. One logical extension is to disentangle latent factors corresponding to more than one known commonality. We also plan to apply our approach in other domains such as speech synthesis.

References

[1] Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Face aging with conditional generative adversarial networks. In *ICIP*, 2017.

[2] David Berthelot, Tom Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv:1703.10717*, 2017.

[3] Jane Bromley. Signature verification using a "siamese" time delay neural network. In *NIPS*, 1994.

[4] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005.

[5] Chris Donahue, Akshay Balsubramani, Julian McAuley, and Zachary C Lipton. Semantically decomposing the latent spaces of generative adversarial networks. *arXiv:1705.07904*, 2017.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

[7] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large scale face recognition. In *ECCV*, 2016.

[8] Rui Huang, Shu Zhang, Tianyu Li, and Ran He. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In *ICCV*, 2017.

[9] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[10] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *SIGIR*, 2015.

[11] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv:1411.1784*, 2014.

[12] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, 2017.

[13] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

[14] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

[15] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, 2017.

[16] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Towards large-pose face frontalization in the wild. In *ICCV*, 2017.

[17] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. In *ICLR*, 2017.

Disentangled Representations for Manipulation of Sentiment in Text

Maria Larsson
Sigma Embedded Engineering, Sweden
maria.larsson@sigma.se

Amanda Nilsson
Zenuity, Sweden
amanda.nilsson@zenuity.com

Mikael Kågebäck
Chalmers University of Technology, Sweden
kageback@chalmers.se

Abstract

The ability to change arbitrary aspects of a text while leaving the core message intact could have a strong impact in fields like marketing and politics by enabling e.g. automatic optimization of message impact and personalized language adapted to the receiver's profile. In this paper we take a first step towards such a system by presenting an algorithm that can manipulate the sentiment of a text while preserving its semantics using disentangled representations. Validation is performed by examining trajectories in embedding space and analyzing transformed sentences for semantic preservation while expression of desired sentiment shift.

1 Introduction

As we live in an increasingly digitized society, algorithms for text analysis and generation can be used for a variety of purposes and may greatly relieve manual work. A system for robust manipulation of global text properties, e.g. sentiment, is one such algorithm that could potentially change how we work with text and open up new possibilities. Though the main purpose of a text might be to communicate a concrete message there are an infinite number of ways the message can be phrased, each with an individual set of global properties connected to it. In this paper we focus on the sentiment aspect and note that robust control over the sentiment would open up a range of new possibilities, like AB testing of different instantiations of a message with respect to some desired measure, and personalized communication automatically adapted to the receiver's profile. Further, the ability of generating new sentences with transformed sentiment could also be useful in data augmentation when the available data is scarce.

Recent work in text generation (Hu et al., 2017; Radford et al., 2017) has shown that it is possible to generate random sentences where the sentiment can be chosen as an input parameter. This line of research has some similarities to the problem we are addressing in this paper but with the key difference that while they generate new random sentences we aim to transform existing sentences. This makes the problem more difficult but also more applicable to real world applications as shown by the recent work of Mueller et al. (2017).

In the visual domain there has been a range of work lately that aims to transform the input image to fit different aspects, e.g. to look like a painting (Gatys et al., 2015). The method presented by Gardner et al. (2015) transforms an image to a deep feature space using a convolutional neural network (CNN). This space is then traversed towards the target features. A new image is subsequently reconstructed from the deep feature representation but where some aspect has been changed from the original image. In their experiments they show that this can be used to transform a smiling portrait into an angry one and make one individual look more like someone else without changing clothing or background. The

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

method we present in this paper is loosely based on their model, however, with significant changes due to the discrete nature of language.

The main contributions of this work include: (1) an algorithm that can automatically transform the sentiment of a text while leaving the semantic content largely intact, and (2) preliminary qualitative analysis of the performance with regard to (a) resulting sentiment, (b) semantic stability and (c) acceptability of the transformed text.

2 Maximum mean discrepancy

The maximum mean discrepancy (MMD) (Gretton et al., 2012) is a test statistic used to determine whether two distributions are the same. Given two distributions, $\mathcal{P}_{\text{source}}$ and $\mathcal{P}_{\text{target}}$, the objective of the MMD is to find a smooth function which is large for samples from $\mathcal{P}_{\text{source}}$ and small for samples from $\mathcal{P}_{\text{target}}$. Given such a function the MMD is the difference between the mean function values for the two sets of samples, which can be empirically estimated as $\text{MMD}(\mathcal{F}, X, Y) =$

$$\sup_{f \in \mathcal{F}} \left(\sum_{i=1}^{m} \frac{f(x_i)}{m} - \sum_{i=1}^{n} \frac{f(y_i)}{n} \right) \tag{1}$$

where $X = [x_1, x_2, \ldots, x_m]$ are samples drawn from the source distribution $\mathcal{P}_{\text{source}}$ and $Y = [y_1, y_2, \ldots, y_n]$ are samples drawn from the target distribution $\mathcal{P}_{\text{target}}$. The function f belongs to a class, $\mathcal{F}$, of smooth functions and should be chosen as to maximize the difference between the mean values of f applied to X and Y. In both (Gretton et al., 2012) and (Gardner et al., 2015), $\mathcal{F}$ is a reproducing kernel Hilbert space allowing comparison of multi-dimensional feature vectors. The function f^* attaining the supremum in equation (1) can be empirically estimated as

$$f^*(z) = \frac{1}{m} \sum_{i=1}^{m} k(x_i, z) - \frac{1}{n} \sum_{i=1}^{n} k(y_i, z), \tag{2}$$

where $k(x, x')$ is a kernel function. The method presented by Gardner et al. (2015) uses a Gaussian kernel function $k(x, x') = e^{-\frac{1}{2\sigma}|x-x'|^2}$ with σ being the kernel bandwidth.

3 Model

The problem we are addressing can be split into three different subtasks. The first task is representing sentences in a continuous space. The second task is exploiting the sentence representation and traversing the manifold in such a way that the sentiment changes. The third task is generating a sentence from the representation space. Our model uses a CNN for sentence encoding. The encoded vectors are subsequently traversed using the MMD statistic and finally decoded using a recurrent neural network (RNN).

3.1 Encoding sentences

A sentence is represented as a matrix where the rows correspond to the, 300-dimensional, *word2vec* (Mikolov et al., 2013) word embeddings for each word in the sentence. This matrix is given as input to a CNN, trained for binary sentiment classification. The network consists of one convolutional layer, one max-pooling layer and finally one fully connected feed forward layer. The filter heights for the convolutional layer are $1, 2, 3$ and 4, and the filter width is 300. 75 filters per size results in a total of 300 filters. The pooling layer therefore outputs a 300-dimensional feature vector, denoted $\mathbf{z}$. This feature vector is extracted from the CNN, along with the predicted label, and used as the encoding of the input sentence.

In addition to classifying sentiment, the CNN needs to encode information about the topic and semantics of the sentence. Therefore, it is trained together with the RNN. Initially, the sentiment classification task is disregarded and the joint networks are trained for encoding and decoding unlabeled sentences. The loss for this task is measured by calculating the cross-entropy error between the predicted word, $\hat{w}$, at position t, in the generated sentence and the actual word, w, at the same position from the original sentence. After this initial training phase, the CNN is trained on binary sentiment classification. The classification loss is calculated as the cross-entropy error between the

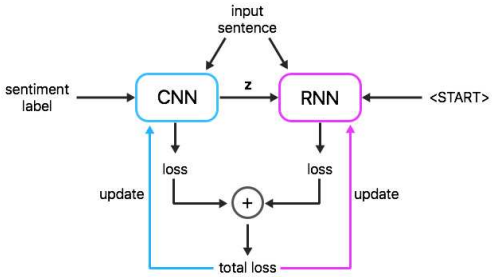


Figure 1: During training, the CNN and RNN are updated using the unweighted sum of the loss for sentiment classification and for text generation.

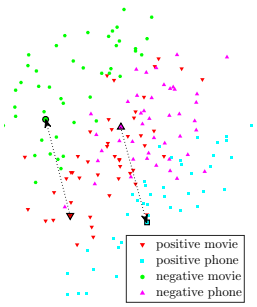


Figure 2: Different icons distinguish feature vectors by sentiment and topic. Bold faced points are examples of original and traversed vectors.

predicted label and the true label for each sentence. This loss is added to the text generation loss, producing a total loss which is used to update the weights in both networks. A schematic of the training procedure is illustrated in figure 1.

3.2 Traversal of the representation space

Since the CNN is trained on binary sentiment classification, two separable distributions of feature vectors are generated. The MMD statistic can be used to traverse a vector originating from one of these distributions to the other. The result of the traversal is a vector that resembles the encoding of a sentence with the opposite sentiment.

When moving the feature vector $\mathbf{z}$ by minimizing equation (2), the semantics of the original sentence may be lost if $\mathbf{z}$ is moved too far along the manifold. To control how far $\mathbf{z}$ is moved from its original position a *budget of change* (Gardner et al., 2015), λ, is used. A source and a target set of sentence representations are created. The source set, $\mathbf{z}^s$, contains feature vectors for sentences with the same sentiment as $\mathbf{z}$ and the target set, $\mathbf{z}^t$, contains feature vectors for sentences with the opposite sentiment. From these sets and the original vector, a matrix $\mathbf{V} = [\mathbf{z}_1^t, \cdots, \mathbf{z}_n^t, \mathbf{z}_1^s, \cdots, \mathbf{z}_m^s, \mathbf{z}]$ is formed. The traversed feature vector can then be expressed as $\mathbf{z}^* = \mathbf{z} + \mathbf{V}\delta$, where δ is a coefficient vector. Equation (2) can now be written as $f^*(\mathbf{z} + \mathbf{V}\delta) = \frac{1}{m}\sum_{i=1}^{m} k(\mathbf{z}_i^s, \mathbf{z} + \mathbf{V}\delta) - \frac{1}{n}\sum_{i=1}^{n} k(\mathbf{z}_i^t, \mathbf{z} + \mathbf{V}\delta)$, where $\delta = \arg\min_\delta f^*(\mathbf{z} + \mathbf{V}\delta) + \lambda\|\mathbf{V}\delta\|^2$, $\lambda \in \mathbb{R}$. The minimization over δ uses the BFGS algorithm (Battiti, 1990) and is constrained by the budget of change, enforced in the last term.

3.3 Decoding sentences

The traversed feature vector $\mathbf{z}^*$ is given as input to an RNN trained for generating text. In addition to $\mathbf{z}^*$, the RNN receives a start-of-sentence token as input in the first time step. For each time step, the RNN outputs the most probable word and gives this word as input to the next time step. When the most probable word is an end-of-sentence token, the generation of words is terminated. The RNN consists of a single layer GRU cell, with a state size of 300. The weight matrix for the input, $\mathbf{W}_x$, consists of the 300-dimensional *word2vec* word embeddings for the words in the vocabulary.

4 Experiments and results

The initial encoding and decoding training uses the large movie review dataset v1.0 (Maas et al., 2011) disregarding the label. The networks are then trained on three sentiment labelled data sets. The first set is the movie review sentence polarity data set v1.0[1] (Pang and Lee, 2005) which consists of 10 662 labelled movie-review sentences from www.rottentomatoes.com. The second set contains 500 reviews for cell phones and accessories from Amazon, 500 reviews for restaurants from Yelp and 500 movie reviews from IMDB[2] (Kotzias et al., 2015). These two sets have equal amounts of positive and

[1]https://www.cs.cornell.edu/people/pabo/movie-review-data/

[2]https://archive.ics.uci.edu/ml/machine-learning-databases/00331/

negative sentences. The third set is a subset of 923 positive and 1320 negative sentences from a data set[3] containing product reviews from various online sources (Täckström and McDonald, 2011). The three data sets are randomly divided 90%-10% into a training and a test set. The training set is used for updating the weights of the networks during training and is divided into batches of 64 sentences. The test set is used for evaluating the accuracy of the networks periodically during training.

4.1 Preserving semantics

In order to evaluate whether the encodings from the CNN contain information about sentiment and semantics, feature vectors for the sentences with different sentiments and topics are visualized. These visualizations also serve as an aid for assessing whether the content in a sentence is preserved in the traversal. The feature vectors are reduced from 300 to 2 dimensions using principal component analysis (PCA) and the visualizations are made using the first two principal components.

The choice of topics was sentences containing either the word *phone* or *movie*, because such sentences would likely have little correlation in contrast to, for example, sentences containing either *comedy* or *drama*. Negative sentences containing the word movie and positive sentences containing the word phone were traversed. The optimization of the MMD was set up with 90 positive examples and 90 negative examples for the source and target sets, and $\lambda = 7\text{e}{-5}$. The examples consisted of an equal amount of sentences containing the word movie and sentences containing the word phone. The topics of the sentences were not used for the traversal but needed when visualizing the results.

The results are shown in figure 2. It is seen that a vector representing a positive sentence containing movie is moved so that the resulting vector lies within the cluster of negative sentences containing movie. In the same way, a vector representing a negative sentence containing phone is moved so that the resulting vector lies within the cluster of positive sentences containing phone. This behaviour suggests that the context and semantics may be preserved during the traversal.

Since the manifold traversal is made using two sets of examples, source and target feature vectors, the traversed feature vector will more resemble the sentences in the target set. This means that if we traverse the manifold for a sentence with a different topic than the sentences in the source and target sets, the traversed vector might not preserve the topic of the original sentence.

Table 1: Regenerated ($\mathbf{z}$), and traversed and generated($\mathbf{z}^*$) sentences compared to the original.

Original:	unfortunately , this is a bad movie that is just plain bad
From $\mathbf{z}$:	unfortunately , this is a bad movie that is just plain bad
From $\mathbf{z}^*$:	overall , this is a good movie that is just good
Original:	one of the oddest and most inexplicable sequels in movie history
From $\mathbf{z}$:	most of the oddest and most strange movie in history history
From $\mathbf{z}^*$:	most interesting and most wonderful movie in one of the oddest ways
Original:	still , i do like this movie for it's empowerment of women there 's not enough movies out there like this one
From $\mathbf{z}$:	still , i do like this movie for one of adults 's not like enough like ages out there 's no women
From $\mathbf{z}^*$:	still , i do not like this movie 's not one of adults for no people who do not like this
Original:	i highly recommend this movie for anyone interested in art , poetry , theater , politics , or japanese history
From $\mathbf{z}$:	i highly recommend this movie , interested for poetry , poetry , poetry , interested in history , or interested history
From $\mathbf{z}^*$:	i highly recommend this movie , except for anything , in any movie , not n't interested in any crappy movie

4.2 Analysis of transformed sentences

There exists no single correct output for the manifold traversal, e.g given the negative sentence "The food did not taste well", both sentences "The food was amazing" and "I liked the food" are valid outputs that reverse the sentiment. Therefore, scores and measures used for other NLP tasks, like BLEU (Papineni et al., 2002) for machine translation, are difficult to apply to the manifold traversal.

[3]https://github.com/oscartackstrom/sentence-sentiment-data

Instead we focus on qualitative evaluation. The encoding-decoding, and the model as a whole, is evaluated by generating sentences from the feature vectors $\mathbf{z}$ (representing the original sentence) and $\mathbf{z}^*$ (the traversed vector) respectively. The generated sentences are manually compared to the original. Ideally, the sentence generated from $\mathbf{z}$ should closely resemble the original sentence while the sentence generated from $\mathbf{z}^*$ should have the same context, but opposite sentiment, as the original sentence. In table 1 some of the better examples of sentences generated by the trained RNN are shown. The overall impression is that, while having poor grammar, the model works well in terms of changing sentiment. We see that the generated sentences have the same topic as the original and that they are composed mostly by the same words. It is also found that shorter sentences are more easily encoded and decoded.

5 Conclusion

An algorithm for sentiment manipulation was presented and evaluated. Visualizations of the embedding space indicate that sentence representations can be moved such that the sentiment changes while the semantics is preserved. Further, examination of generated sentences from manipulated embeddings confirmed that the sentiment had changed while the semantics and acceptability had stayed largely constant.

Acknowledgments

The authors would like to acknowledge the project *Towards a knowledge-based culturomics* supported by a framework grant from the Swedish Research Council (2012–2016; dnr 2012-5738).

References

Roberto Battiti. 1990. Optimization methods for back-propagation: Automatic parameter tuning and faster convergence. In *International Joint Conference on Neural Networks*. volume 1, pages 593–596.

Jacob R. Gardner, Matt J. Kusner, Yixuan Li, Paul Upchurch, Kilian Q. Weinberger, and John E. Hopcroft. 2015. Deep manifold traversal: Changing labels with convolutional features. *CoRR* abs/1511.06421.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2015. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576* .

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *Journal of Machine Learning Research* 13(Mar):723–773.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Controllable text generation. *arXiv preprint arXiv:1703.00955* .

Dimitrios Kotzias, Misha Denil, Nando de Freitas, and Padhraic Smyth. 2015. From group to individual labels using deep features. In *KDD*. ACM, pages 597–606.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, pages 142–150.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.

Jonas Mueller, David Gifford, and Tommi Jaakkola. 2017. Sequence to better sequence: continuous revision of combinatorial structures. In *International Conference on Machine Learning*. pages 2536–2544.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*. pages 115–124.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '02, pages 311–318. https://doi.org/10.3115/1073083.1073135.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. Cite arxiv:1704.01444.

Oscar Täckström and Ryan McDonald. 2011. Discovering fine-grained sentiment with latent variable structured prediction models. In *Proceedings of the 33rd European Conference on Advances in Information Retrieval*. Springer-Verlag, Berlin, Heidelberg, ECIR'11, pages 368–374.

Disentangling the independently controllable factors of variation by interacting with the world

Valentin Thomas[*1,2] Emmanuel Bengio[*3] William Fedus[*1] Jules Pondard[4]
Philippe Beaudoin[2] Hugo Larochelle[1,5] Joelle Pineau[3,6] Doina Precup[3,7]
Yoshua Bengio[1,8]

Abstract

It has been postulated that a good representation is one that disentangles the underlying explanatory factors of variation. However, it remains an open question what kind of training framework could potentially achieve that. Whereas most previous work focuses on the static setting (e.g., with images), we postulate that some of the causal factors could be discovered if the learner is allowed to interact with its environment. The agent can experiment with different actions and observe their effects. More specifically, we hypothesize that some of these factors correspond to aspects of the environment which are independently controllable, i.e., that there exists a policy and a learnable feature for each such aspect of the environment, such that this policy can yield changes in that feature with minimal changes to other features that explain the statistical variations in the observed data. We propose a specific objective function to find such factors, and verify experimentally that it can indeed disentangle independently controllable aspects of the environment without any extrinsic reward signal.

1 Introduction

When solving Reinforcement Learning problems, what separates great results from random policies is often having the right feature representation. Even with function approximation, learning the right features can lead to faster convergence than blindly attempting to solve given problems [Jaderberg et al., 2016].

The idea that learning good representations is vital for solving most kinds of real-world problems is not new, both in the supervised learning literature [Bengio, 2009, Goodfellow et al., 2016], and in the RL literature [Dayan, 1993, Precup, 2000]. An alternate idea is that these representations do not need to be learned explicitly, and that learning can be guided through internal mechanisms of reward, usually called intrinsic motivation [Barto et al., Oudeyer and Kaplan, 2009, Salge et al., 2013].

We build on a previously studied [Thomas et al., 2017] mechanism for representation learning that has close ties to intrinsic motivation mechanisms and causality. This mechanism explicitly links the agent's control over its environment to the representation of the environment that is learned by the agent. More specifically, this mechanism's hypothesis is that most of the underlying factors of variation in the environment can be controlled by the agent independently of one another.

We propose a general and easily computable objective for this mechanism, that can be used in any RL algorithm that uses function approximation to learn a latent space. We show that our mechanism can push a model to learn to disentangle its input in a meaningful way, and learn to represent factors

[*]Equal contribution, random order, [1]MILA, Université de Montréal, [2]Element AI, [3]McGill University, [4]ENS Paris, [5]Google Brain, [6]Facebook AI Research, [7]Google Deepmind, [8]CIFAR Senior Fellow

Learning Disentangled Representations, NIPS 2017 Workshop. Long Beach, CA, USA.

which take multiple actions to change and show that these representations make it possible to perform model-based predictions in the learned latent space, rather than in a low-level input space (e.g. pixels).

2 Learning disentangled representations

The canonical deep learning framework to learn representations is the autoencoder framework [Hinton and Salakhutdinov, 2006]. There, an encoder $f : S \to H$ and a decoder $g : H \to S$ are trained to minimize the *reconstruction error*, $\|s - g(f(s))\|_2^2$. H is called the latent (or representation) space, and is usually constrained in order to push the autoencoder towards more desirable solutions. For example, imposing that $H \in \mathbb{R}^K, S \in \mathbb{R}^N,\ K \ll N$ pushes f to learn to compress the input; there the bottleneck often forces f to extract the principal factors of variation from S. However, this does not necessarily imply that the learned latent space disentangles the different factors of variations. Such a problem motivates the approach presented in this work.

Other authors have proposed mechanisms to disentangle underlying factors of variation. Many deep generative models, including variational autoencoders [Kingma and Welling, 2014] , generative adversarial networks [Goodfellow et al., 2014] or non-linear versions of ICA [Dinh et al., 2014, Hyvarinen and Morioka, 2016] attempt to disentangle the underlying factors of variation by assuming that their joint distribution (marginalizing out the observed s) factorizes, i.e., that they are marginally independent.

Here we explore another direction, trying to exploit the ability of a learning agent to act in the world in order to impose a further constraint on the representation. We hypothesize that interactions can be the key to learning how to disentangle the various causal factors of the stream of observations that an agent is faced with, and that such learning can be done in an unsupervised way.

3 The selectivity objective

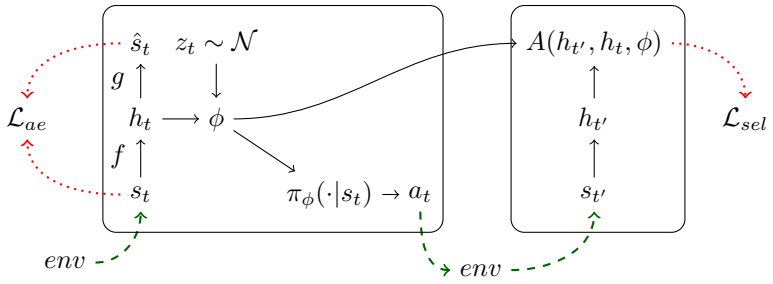
We consider the classical reinforcement learning setting but in the case where extrinsic rewards are not available. We introduce the notion of **controllable factors of variation** $\phi \in \mathbb{R}^K$ which are generated from a neural network $\Phi(h, z), z \sim \mathcal{N}(0, 1)^m$ where $h = f(s)$ is the current latent state. The factor ϕ represents an embedding of a policy π_ϕ whose goal is to realize the variation ϕ in the environment.

To discover meaningful factors of variation ϕ and their associated policies π_ϕ, we consider the following general quantity $\mathcal{S}$ which we refer to as selectivity and that is used as a reward signal for π_ϕ:

$$\mathcal{S}(h, \phi) = \mathbb{E}\left[\log \frac{A(h', h, \phi)}{\mathbb{E}_{p(\varphi|h)}[A(h', h, \varphi)]} \;\middle|\; s' \sim \mathcal{P}_{ss'}^{\pi_\phi}\right] \tag{1}$$

Here $h = f(s)$ is the encoded initial state before executing π_ϕ and $h' = f(s')$ is the encoded terminal state. ϕ and φ represent factors of variation a *factor*. $A(h', h, \phi)$ should be understood as a score describing how close ϕ is to the variation it caused in (h', h). For example in the experiments of section 4.1, we choose A to be a gaussian kernel between $h' - h$ and ϕ, while in the experiments of section 4.2, we choose $A(h', h, \phi) = \max\{0, \langle h' - h, \phi\rangle\}$. The intuition behind these objectives is that in expectation, a factor ϕ should be close to the variation it caused (h', h) when following π_ϕ compared to other factors φ that could have been sampled and followed thus encouraging **independence** within the factors.

Conditioned on a scene representation h, a distribution of policies are feasible. Samples from this distribution represent ways to modify the scene and thus may trigger an internal selectivity reward signal. For instance, h might represent a room with objects such as a light switch. $\phi = \phi(h, z)$ can be thought of as the distributed representation for the "name" of an underlying factor, to which is associated a policy and a value. In this setting, the light in a room could be a factor that could be either on or off. It could be associated with a policy to turn it on, and a binary value referring to its state, called an attribute or a feature value.We wish to jointly learn the policy $\pi_\phi(\cdot|s)$ that modifies the scene, so as to control the corresponding value of the attribute in the scene, whose variation is computed by a scoring function $A(h', h, \phi) \in \mathbb{R}$. In order to get a distribution of such embeddings, we compute $\phi(h, z)$ as a function of h and some random noise z.



(a)

Figure 1: The computational model of our architecture. s_t is the first state, from its encoding h_t and a noise distribution z, ϕ is generated. ϕ is used to compute the policy π_ϕ, which is used to act in the world. The sequence $h_t, h_{t'}$ is used to update our model through the selectivity loss, as well as an optional autoencoder loss on h_t.

The goal of a selectivity-maximizing model is to find the density of factors $p(\phi|h)$, the latent representation h, as well as the policies π_ϕ that maximize $\mathbb{E}_{p(\phi|h)}[\mathcal{S}(h, \phi)]$.

3.1 Link with mutual information and causality

The selectivity objective, while intuitive, can also be related to information theoretical quantities defined in the latent space. From [Donsker and Varadhan, 1975, Ruderman et al., 2012] we have $\mathcal{D}_{\text{KL}}\big(p||q\big) = \sup_{A \in \mathcal{L}^\infty(q)} \mathbb{E}_p[\log A] - \log \mathbb{E}_q[A]$. Applying this equality to the mutual information $\mathcal{I}_p(\phi, h'|h) = \mathbb{E}_{p(h'|h)}\big[\mathcal{D}_{\text{KL}}\big(p(\phi|h', h)||p(\phi|h)\big)\big]$ gives

$$\mathcal{I}_p(\phi, h'|h) \geq \sup_\theta \mathbb{E}_{p(\phi|h)}\big[\mathcal{S}(h, \phi)\big]$$

where θ is the set of weights shared by the factor generator, the policy network and the encoder.

Thus, our total objective along entire trajectories is a lower bound on the causal [Ziebart, 2010] or directed [Massey, 1990] information $\mathcal{I}_p(\phi \mapsto h) = \sum_t \mathcal{I}_p(\phi_t, h_t|h_{t-1})$ which is a measure of the **causality** the process ϕ exercises on the process h. See Appendix C for details.

4 Experiments

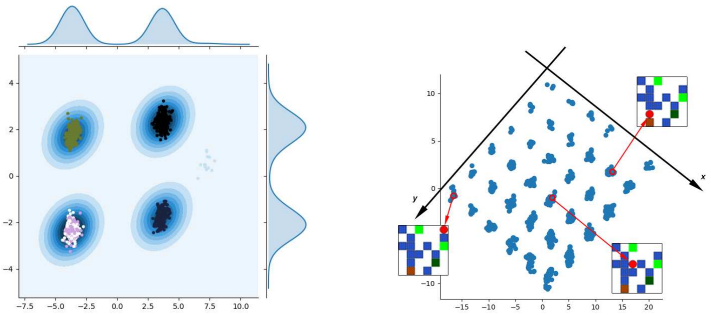
We use MazeBase [Sukhbaatar et al., 2015] to assess the performance of our approach. We do not aim to solve the game. In this setting, the agent (a red circle) can move in a small environment (64×64 pixels) and perform the actions `down, left, right, up`. The agent can go anywhere except on the orange blocks.

4.1 Learned representations

After jointly training the reconstruction and selectivity losses, our algorithm disentangles four directed factors of variations as seen in Figure 2: $\pm x$-position and $\pm y$-position of the agent. For visualization purposes we chose the bottleneck of the autoencoder to be of size $K = 2$. To complicate the disentanglement task, we added the redundant action `up` as well as the action `down+left` in this experiment.

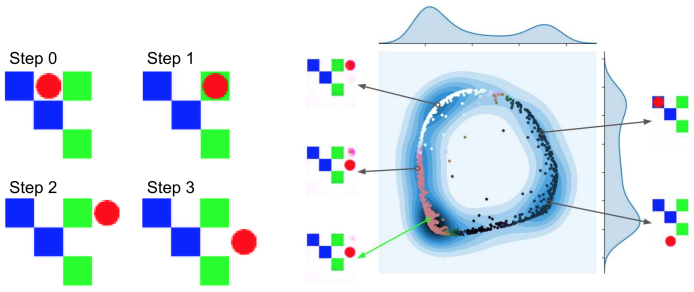
The disentanglement appears clearly as the latent features corresponding to the x and y position are orthogonal in the latent space. Moreover, we notice that our algorithm assigns both actions `up` (white and pink dots in Figure 2.a) to the same feature. It also does not create a significant mode for the feature corresponding to the action `down+left` (light blue dots in Figure 2.a) as this feature is already explained by features `down` and `left`.

[1] pink and white for `up`, light blue for `down+left`, green for `right`, purple black `down` and night blue for `left`.



(a) (b)

Figure 2: (a) Sampling of 1000 variations $h' - h$ and its kernel density estimation encountered when sampling random controllable factors ϕ. We observe that our algorithm disentangles these representations on 4 main modes, each corresponding to the action that was actually taken by the agent.[1] (b) The disentangled structure in the latent space. The x and y axis are disentangled such that we can recover the x and y position of the agent in any observation s simply by looking at its latent encoding $h = f(s)$. The missing point on this grid is the only position the agent cannot reach as it lies on an orange block.



(a) (b)

Figure 3: (a) The actual 3-step trajectory done by the agent. (b) PCA view of the space $\phi(h_0, z), z \sim \mathcal{N}(0, 1)$. Each arrow points to the reconstruction of the prediction $T_\theta(h_0, \phi)$ made by different ϕ. The ϕ at the start of the green arrow is the one used by the policy in (a). Notice how its prediction accurately predicts the actual final state.

4.2 Multistep embedding of policies

In this experiment, ϕ are embeddings of 3-steps policies π_ϕ. We add a model-based loss $\mathcal{L}_{MB} = ||h_{t+3} - T_\theta(h_t, \phi)||^2$ defined only in the latent space, and jointly train a decoder alongside with the encoder. Notice that we never train our model-based cost at pixel level. While we currently suffer from mode collapsing of some factors of variations, we show that we are successfully able to do predictions in latent space, reconstruct the latent prediction with the decoder, and that our factor space disentangles several types of variations.

5 Conclusion, success and limitations

Pushing representations to model independently controllable features currently yields some encouraging success. Visualizing our features clearly shows the different controllable aspects of simple environments, yet, our learning algorithm is unstable. What seems to be the strength of our approach could also be its weakness, as the independence prior forces a very strict separation of concerns in the learned representation, and should maybe be relaxed.

Some sources of instability also seem to slow our progress: learning a conditional distribution on controllable aspects that often collapses to fewer modes than desired, learning stochastic policies that

often optimistically converge to a single action, tuning many hyperparameters due to the multiple parts of our model. Nonetheless, we are hopeful in the steps that we are now taking. Disentangling happens, but understanding our optimization process as well as our current objective function will be key to further progress.

References

Andrew G Barto, Satinder Singh, and Nuttapong Chentanez. Intrinsically motivated learning of hierarchical collections of skills.

Yoshua Bengio. *Learning deep architectures for AI*. Now Publishers, 2009.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear Independent Components Estimation. arXiv:1410.8516, ICLR 2015 workshop, 2014.

Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time, i. *Communications on Pure and Applied Mathematics*, 28(1):1–47, 1975.

Carlos Florensa, Yan Duan, and Pieter Abbeel. Stochastic neural networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1704.03012*, 2017.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. In *NIPS'2014*, 2014.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Aapo Hyvarinen and Hiroshi Morioka. Unsupervised Feature Extraction by Time-Contrastive Learning and Nonlinear ICA. In *NIPS*, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016.

Durk P. Kingma and Max Welling. Auto-encoding variational Bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

Yann LeCun. The MNIST database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*, 1998.

James Massey. Causality, feedback and directed information. In *Proc. Int. Symp. Inf. Theory Applic.(ISITA-90)*, pages 303–305, 1990.

Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1:6, 2009.

Doina Precup. Temporal abstraction in reinforcement learning. 2000.

Avraham Ruderman, Mark Reid, Darío García-García, and James Petterson. Tighter variational representations of f-divergences via restriction to probability measures. *arXiv preprint arXiv:1206.4664*, 2012.

Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment - an introduction. *CoRR*, abs/1310.1863, 2013. URL `http://arxiv.org/abs/1310.1863`.

Sainbayar Sukhbaatar, Arthur Szlam, Gabriel Synnaeve, Soumith Chintala, and Rob Fergus. MazeBase: A sandbox for learning from games. *arXiv preprint arXiv:1511.07401*, 2015.

Valentin Thomas, Jules Pondard, Emmanuel Bengio, Marc Sarfati, Philippe Beaudoin, Marie-Jean Meurs, Joelle Pineau, Doina Precup, and Yoshua Bengio. Independently controllable factors. *CoRR*, abs/1708.01289, 2017. URL `http://arxiv.org/abs/1708.01289`.

Brian D Ziebart. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University, 2010.

A Additional details

A.1 Architecture

Our architecture is as follows: the encoder, mapping the raw pixel state to a latent representation, is a 4-layer convolutional neural network with batch normalization [Ioffe and Szegedy, 2015] and leaky ReLU activations. The decoder uses the transposed architecture with ReLU activations. The noise z is sampled from a 2-dimensional gaussian distribution and both the generator $\Phi(h, z)$ and the policy $\pi(h, \phi)$ are neural networks consisting of 2 fully-connected layers. In practice, a minibatch of $n = 256$ or 1024 vectors $\phi_1, \ldots, \phi_n$ is sampled at each step. The agent randomly choses one $\phi = \phi_{behavior}$ and samples actions from its policy $a \sim \pi(h, \phi_{behavior})$. Our model parameters are then updated using policy gradient with the REINFORCE estimator and a state-dependent baseline and importance sampling. For each selectivity reward, the term $\mathbb{E}_{\phi'}[A(h', h, \phi')]$ is estimated as $\frac{1}{n}\sum_{i=1}^{n} A(h', h, \phi_i)$.

In practice, we don't use concatenation of vectors when feeding two vectors as input for a network (like (h, z) for the factor generator or (h, ϕ) for the policy). For vectors $a, b \in \mathbb{R}^{n_a \times n_b}$. We use a bilinear operation $bil(a, b) = (a_i * b_j)_{i \in [[n_a]], j \in [[n_b]]}$ as in Florensa et al. [2017]. We observe the bilinear integrated input to more strongly enforce dependence on both vectors; in contrast, our models often ignored one input when using a simple concatenation.

Through our research, we experiment with different outputs for our generator $\Phi(h, z)$. We explored embedding the ϕ-vectors into a hypercube, a hypersphere, a simplex and also a simplex multiplied by the output of a $tanh(\cdot)$ operation on a scalar.

A.2 First experiment

In the first experiment, figure 2, we used a gaussian similarity kernel i.e $A(h', h, \phi) = \exp(-\frac{||h'-(h+\phi)||^2}{2\sigma^2})$ with $\sigma = \sqrt{dim(h)}$. In this experiment only, for clarity of the figure, we only allowed permissible actions in the environment (no no-op action).

B Additional Figures

B.1 Discrete simple case

Here we consider the case where we learn a latent space H of size K, with K factors corresponding to the coordinates of h ($h_i,\ i \in [k]$), and learn K separately parameterized policies $\pi_i(a|h),\ i \in [k]$. We train our model with the selectivity objective, but no autoencoder loss, and find that we correctly recover independently controllable features on a simple environment. Albeit slower than when jointly training an autoencoder, this shows that the objective we propose is strong enough to provide a learning signal for discovering a disentangled latent representation.

We train such a model on a gridworld MNIST environment, where there are two MNIST digits . The two digits can be moved on the grid via 4 directional actions (so there are 8 actions total), the first digit is always odd and the second digit always even, so they are distiguishable. In Figure 4 we plot each latent feature h_k as a curve, as a function of each ground truth. For example we see that the black feature recovers $+x_1$, the horizontal position of the first digit, or that the purple feature recovers $-y_2$, the vertical position of the second digit.

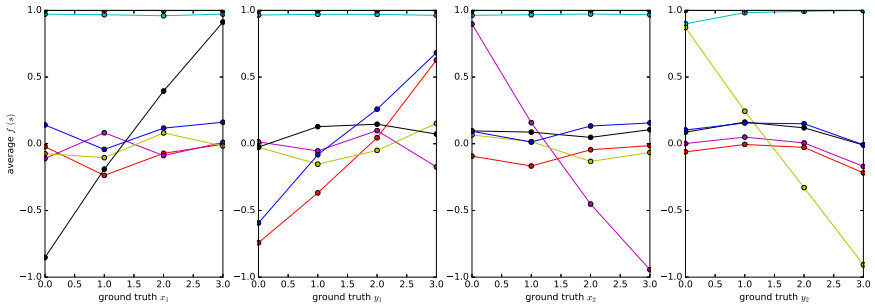


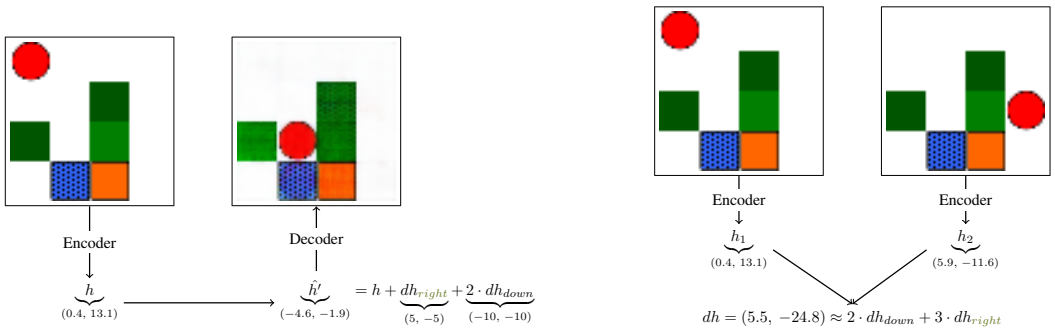
Figure 4: In a gridworld environment with 2 objects (in this case 2 MNIST digits), we know there are 4 underlying features, the (x_i, y_i) position of each digit i. Here each of the four plots represents the evolution of the f_k's as a function of their underlying feature, from left to right x_1, y_1, x_2, y_2. We see that for each of them, at least one f_k recovers it almost linearly, from the raw pixels only.

B.2 Planning and policy inference example in 1-step

This disentangled structure could be used to address many challenging issues in reinforcement learning. We give two examples in figure 5:

- Model-based predictions: Given an initial state, s_0, and an action sequence $a_{\{0:T-1\}}$, we want to predict the resulting state s_T.
- A simplified deterministic policy inference problem: Given an initial state s_{start} and a terminal state s_{goal}, we aim to find a suitable action sequence $a_{\{0:T-1\}}$ such that s_{goal} can be reached from s_{start} by following it.

Because of the $tanh$ activation on the last layer of $\Phi(h, z)$, the different factors of variation $dh = h' - h$ are placed on the vertices of a hypercube of dimension K, and we can think of the the policy inference problem as finding a path in that simpler space, where the starting point is h_{start} and the goal is h_{goal}. We believe this could prove to be a much easier problem to solve.

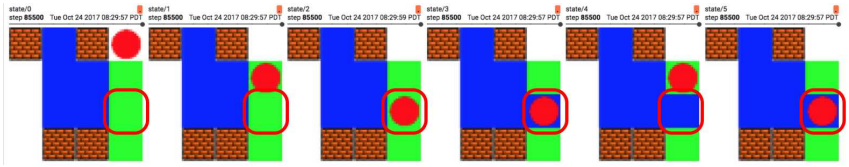


(a) (b)

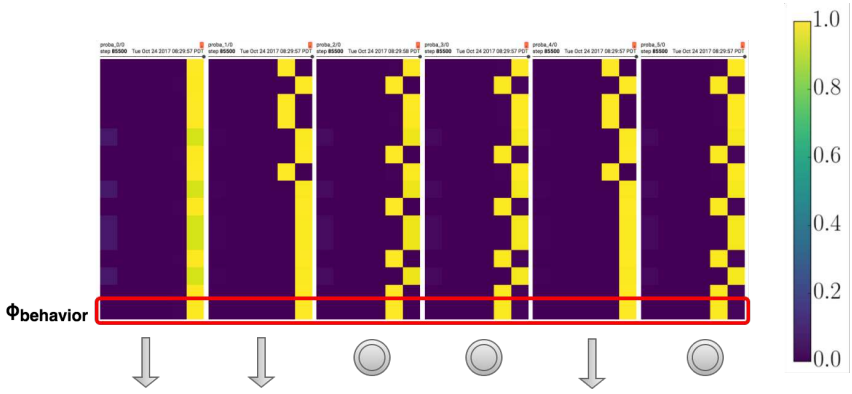
Figure 5: (a) Predicting the effect of a cause on Mazebase. The leftmost image is the visual input of the environment, where the agent is the round circle, and the switch states are represented by shades of green. After the training, we are able to distinguish one cluster per dh (Figure 2), that is to say per variation obtained after performing an action, independently from the position h. Therefore, we are able to move the agent just by adding the corresponding dh to our latent representation h. The second image is just the reconstruction obtained by feeding the resulting h' into the decoder. (b) Given a starting state and a goal state, we are able to decompose the difference of the two representations dh into a (non-directed) sequence of movements.

B.3 Multistep Example

We demonstrate an instance of ICF operating in a 4×4 Mazebase enviroment over five time steps in Figure 6. We consistently witness a failure of mode collapse in our generator Φ and therefore the generator only produces a subset of all possible ϕ-variations. In Figure 6, we observe the ϕ governing the agent's policy π_ϕ appears to correspond to moving two positions down and then to repeatedly toggle the switch. A random action due to ϵ-greedy led to the agent moving up and off the switch at time step-4. This perturbation is corrected by the policy π_ϕ by moving down in order to return to toggling the relevant switch.



(a)



(b)

Figure 6: (a) Mazebase environment over five time-steps. Here the red dot denotes the position of the agent. The $\phi_{behavior}$ governing the agent's policy appears to control toggling the switch indicated by the red rounded box. (b) Visualization of the policies instantiated by different ϕs. Each box represents the probability distribution of the policies at that time step. Each row is generated by a different ϕ and each column corresponds to an action (up, left, pass, right, toggle, down) in order. The boxed column shows the $\phi_{behavior}$. The symbols below each box represent the most-probable action for the behavioral policy, where the grey circle indicates toggling the switch.

C Variational bound and the selectivity

Let us call $p(h_{t+1}|\phi_{t+1}, h_t) = \mathcal{P}^{\phi}_{h',h}$ the probability distribution over final hidden states starting from h and using the policy parametrized by the embedding ϕ.

$p(h_{t+1}|\phi_{t+1}, h_t) = \Pi_{k=1}^{K} \pi_{\phi_{t+1}}(a_{t+\frac{k-1}{K}}|h_{t+\frac{k-1}{K}}) p_{env}(s_{t+\frac{k}{K}}|a_{t+\frac{k-1}{K}}, s_{t+\frac{k-1}{K}})$. $p(\bullet|\phi, h)$

For simplicity, let's refer to h_t as h, h_{t+1} as h' and ϕ_{t+1} as ϕ

C.1 Lower bound on the mutual information

In the case where $A(h', h, \phi)$ is actually is probability density $q(h'|h, \phi)$ (or any unnormalized density such that the normalization factor does not depend on ϕ) we have:

$$\begin{aligned}
\mathcal{S}(\phi, h) &= \mathbb{E}_{h' \sim p(h'|\phi,h)} \log \frac{q(h'|\phi, h)}{\mathbb{E}_{\varphi|h} q(h'|\varphi, h)} \\
&= \mathbb{E}_{\phi|h}\big[\mathcal{D}_{\mathrm{KL}}\big(p(h'|\phi, h)||q(h'|h)\big) - \mathcal{D}_{\mathrm{KL}}\big(p(h'|\phi, h)||q(h'|\phi, h)\big)\big] \\
&= \mathbb{E}_{\phi|h}\big[\mathcal{D}_{\mathrm{KL}}\big(p(h'|\phi, h)||q(h'|h)\big)\big] - \mathcal{D}_{\mathrm{KL}}\big(p(h'|h)||q(h'|h)\big) - \mathbb{E}_{p(h'|h)}\big[\mathcal{D}_{\mathrm{KL}}\big(p(\phi|h', h)||q(\phi|h', h)\big)\big] \\
&= \mathbb{E}_{\phi|h}\big[\mathcal{D}_{\mathrm{KL}}\big(p(h'|\phi, h)||p(h'|h)\big)\big] - \mathbb{E}_{p(h'|h)}\big[\mathcal{D}_{\mathrm{KL}}\big(p(\phi|h', h)||q(\phi|h', h)\big)\big] \\
&= \mathcal{I}^p(\phi, h'|h) - \mathbb{E}_{p(h'|h)}\big[\mathcal{D}_{\mathrm{KL}}\big(p(\phi|h', h)||q(\phi|h', h)\big)\big]
\end{aligned}$$

where we used that fact that $p(\phi|h) = q(\phi|h)$ and by design we only sample ϕs from one generator.

Thus, the gap between the selectivity and the mutual information is the KL divergence of the two posterior distributions.

As we sample the factors ϕ uniformly, our total objective is then

$$\mathcal{J}(\theta) = \mathcal{I}(\phi^T \mapsto h^T) - \sum_t \mathbb{E}_{p(h_{t+1}|h_t)}\big[\mathcal{D}_{\mathrm{KL}}\big(p(\phi_{t+1}|h_{t+1}, h_t)||q(\phi_{t+1}|h_{t+1}, h_t)\big)\big]$$

where $\mathcal{I}^p(\phi \mapsto h)$ is often refered as the *directed information* [Massey, 1990] Ziebart [2010]

The bound

$$\mathcal{I}_p(\phi, h'|h) \geq \sup_\theta \mathbb{E}_{p(\phi|h)}\big[\mathcal{S}(h, \phi)\big]$$

is true even when A is not a probability density and can be proven directly by using the equality from [Donsker and Varadhan, 1975, Ruderman et al., 2012]

$$\mathcal{D}_{\mathrm{KL}}\big(p||q\big) = \sup_{T \in \mathcal{L}^\infty(q)} \mathbb{E}_p\big[T\big] - \log \mathbb{E}_q\big[e^T\big]$$

for $T = \log A$ and using the identity $\mathcal{I}_p(X, Y) = \mathbb{E}_{p(y)}\big[\mathcal{D}_{\mathrm{KL}}\big(p(x|y)||p(x)\big)\big]$

D Additional information on the training

In our experiments, we use the selectivity objective, an autoencoding loss and an entropy regularization loss $\mathcal{H}(\pi_\phi)$ for each of the policies π_ϕ. Furthermore, in experiment 4.2 we added the model-based cost $||h' - T(h, \phi)||^2$ with T a learned two layer fully connected neural network.

The selectivity is used to update the parameters of the encoder, factor generator and policy networks. We use the following equation for computing the gradients

$$\nabla_\theta \mathbb{E}_{\pi_\theta}\big[f_\theta\big] = \mathbb{E}_{\pi_\theta}\big[\nabla_\theta f_\theta + f_\theta \nabla_\theta \log \pi_\theta\big]$$

We also use a state dependent baseline V as a control variate to reduce the variance of the REINFORCE estimator.

Furthermore, to be able to train the factor generator efficiently, we train all ϕ sampled in a mini-batch (of size 1024) by importance sampling on the probability ratio of the trajectory under each ϕ