BACK TRANSLATION VARIATIONAL AUTOENCODERS FOR OOD GENERATION

Frantzeska Lavda[1,2] **Alexandros Kalousis**[2]

[1]Computer Science Department, University of Geneva
[2]Geneva School of Business Administration,
University of Applied Sciences and Arts Western Switzerland (HES-SO)
frantzeska.lavda@hesge.ch, alexandros.kalousis@hesge.ch

ABSTRACT

Humans are able to quickly adapt to new situations, learn effectively with limited data, and create unique combinations of basic concepts. In contrast generalizing out-of-distribution (OOD) data and achieving combinatorial generalizations are fundamental challenges for the machine learning models. To address these challenges, we propose BtVAE, a method that employs supervised conditional VAE models to achieve combinatorial generalization in certain scenarios and consequently to generate out-of-distribution (OOD) data. Unlike previous approaches that use new factors of variation during testing, our method uses only existing attributes from the training data, but in ways that were not seen during training (e.g., small objects during training and large objects during testing).

We first learn a latent representation of the in-distribution inputs and we passing this representation in a conditional decoder, conditioning on some OOD attribute values, to generate implicit OOD samples. These generated samples are then translated back to the original in-distribution inputs, conditioning on the actual attribute values. To ensure that the generated OOD samples have the specified OOD attribute values, a predictor is introduced. By training with OOD attribute values the decoder learns to produce the correct output for unseen combinations, resulting in a model that not only is able to reconstruct OOD data but also to manipulate the OOD data and to generate samples conditioning on unseen combinations of attribute values.

1 INTRODUCTION

Combinatorial generalization, the ability to understand and produce novel combinations of familiar elements, is a key aspect of human intelligence. Humans can make ”infinite use of finite means” (Wilhelm Von Humboldt, 1999; Chomsky, 2014), using a small set of elements (such as words) to create limitless combinations (such as new sentences) (Battaglia et al., 2018). For example, one can imagine a pink elephant even if they have never seen one before. While color and object are independent, for a human brain imagining a pink elephant is a trivial task. However, for machine learning (ML) models generating a pink elephant is not as straightforward if there are no pink elephant in the training data as they struggle generating OOD data or mixing existing attributes (color and object) (Lake et al., 2017; Bengio et al., 2018). Based on Battaglia et al. (2018) combinatorial generalization should be one of the top priorities in modern artificial intelligence.

In this paper, we propose Back translation VAE (BtVAE) for conditionally generating OOD data. Our aim is to 1. reconstruct missing combinations of data, 2. manipulate the attributes of the OOD data and 3. generate samples conditioned on previously unseen combinations of properties. This can be seen as OOD generation as we aim to generate unseen data. Notably, the OOD data that we assume do not constitute new factors of variation, such as new attributes or domains, but combinations of attributes that were not previously observed during training.

OOD generation is a relatively new field of research. Data augmentation is an effective way to increase data diversity and it can therefore improve OOD generalization (Xie et al., 2020). Lee et al.

(2018), Sricharan & Srivastava (2018), Vernekar et al. (2019) uses GANs and/or VAEs to generate out-of-distribution samples in order to improve OOD detection. They do not focus at the capability of the model to generate OOD data, their goal is to augment the training set with OOD data in order to learn better the OOD classifier for OOD detection. Their resulting OOD samples mimic in-distribution samples or are confined to the boundary of in-distribution samples. Unlike these works, our approach focuses solely on generating OOD samples. Additionally, our approach focuses on generating data from combinations of attributes that have never been seen during training, and not from the boundary of in-distribution samples.

Recently, several papers using similar OOD dataset as in this paper explore whether models that have high disentanglement performance are also able to perform certain forms of combinatorial generalization. (Higgins et al., 2018; Watters et al., 2019; Dittadi et al., 2021; Montero et al., 2021; Montero et al.; Schott et al., 2022). Dittadi et al. (2021); Cai et al. (2019) have shown promising results using disentanglement models for OOD tasks but the models have been tested on simple OOD data where only a small number of combinations were excluded Montero et al. (2021). In contrast recent studies such as Montero et al. (2021); Montero et al.; Schott et al. (2022) have tested different models under more challenging conditions found no evidence that the disentanglement representation supports combinatorial generalization in both latent space and reconstruction space under challenging generalisation conditions (larger number of combinations are excluded from the training set). In this paper we test the OOD generation performance of our model by excluding certain combinations of attribute values from the training data, similar to the approach used in Montero et al. (2021); Montero et al.; Schott et al. (2022) but instead of aiming for a disentangled latent space we aim conditional generation of unseen properties.

In this paper, we propose Back Translation VAE (BtVAE), a conditional generative learning framework which aims to achieve combinatorial generalization. Our approach uses a VAE with a conditional decoder that learns to reconstruct and to modify OOD data that have the same attributes as the training data but in combinations that have never been seen during training. To accomplish this, we randomly sample values for the conditioning attributes and pass them to the decoder. This ensures that the decoder receives combinations of attributes that may not exist in the training data. Finally, we pass the output of the decoder through a second VAE (with shared parameters) conditioning this time to the corresponding attribute values of the in-distribution input. This way, we translate the OOD generated data back to the in-distribution data and can check the reconstruction error of our model. Our translation procedure can be seen as a cycle consistency constraint Zhu et al. (2017); Kim et al. (2017); Jha et al. (2018), but it doesn't require swapping the latent representations of two different

2 BtVAE

In this section, we present BtVAE, a generative model for OOD generation. Our goal is to learn a conditional generation model $p(\mathbf{x}|\mathbf{y})$ that will allow us

1. to generate data conditioning on attribute values combinations that have never been observed in the training set.
2. to reconstruct OOD data.
3. to manipulate certain desired characteristics of the OOD data before reconstructing it.

2.1 Problem formulation

We want to learn a conditional generation model that learns mapping among unseen combination of attributes that will allows us to generate samples conditioning on desired unseen combinations of attribute values that are not included at the training data. As in real-world applications, we only have access to a finite set of inputs and attributes and the data often are collected from two related but distinct distributions. So, we consider an OOD setting where the prior distribution $p_{train}(y)$ during training is different from the prior distribution $p_{test}(y)$ during testing. Hence, some values in the set $\boldsymbol{A}^b$ of an attribute y^b are not present in p_{train} but are in p_{test}:

$$p_{train}(y^b = v) = 0 \qquad\qquad p_{test}(y^b = v) > 0 \tag{1}$$

In fact, the training and test distributions are completely disjoint, meaning that each point can only have non-zero probability mass in either p_{train} or p_{test}.

For example, if the dataset contains images of people's face, the training dataset does not contain images of women wearing glasses or images of men with blond hair, if the dataset is a set of molecules, the training dataset might not contain a range of LogP or molecular weight values and the test dataset consist only from these excluded combinations. We have to mention that while p_{train} and p_{test} are different on our setting, they are both related to the true distribution p.

We consider a conditional latent-variable model $p_\theta(\mathbf{x}, \mathbf{z}|\mathbf{y}) = p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z})p(\mathbf{z})$, where $\mathbf{x}$ denotes an observation, $\mathbf{y}$ represents the associated attributes and $\mathbf{z}$ the associated latent variables. The marginal $p(\mathbf{z})$ is a prior over the latent variable and $p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z})$ is an exponential family distribution whose natural parameter is a function of $\mathbf{z}$ parametrized by θ, e.g. through a neural network. A generative conditional model may not always effectively use the conditioning information during the generative process, particularly when dealing with out-of-distribution data. To address this issue and improve OOD generations, we propose BtVAE which employs a conditional VAE and the back translation procedure. During the training the input observation is modified by conditioning it on randomly chosen attribute values, and then the modified input is used to reconstruct the original observation while conditioning on the actual attributes. By conditioning on randomly chosen attribute values during training, the model learns to handle a variety of combinations of input-attribute pairs, that may are excluded from the training data thus making it capable of handling OOD data.

2.2 Model

BtVAE, Figure 1, is composed of a probabilistic encoder and a conditional probabilistic decoder. The encoder maps an input data point $\mathbf{x}$ to its latent representation $\mathbf{z}$. The decoder then takes this latent representation and random sampled attribute values from the prior, $\tilde{\mathbf{y}} \sim p(\mathbf{y})$ and generates a new version of the input, $\tilde{\mathbf{x}}$, that may have attribute value combinations that do not exist in the training data. In the same time we aim to keep the other details of the given input $\mathbf{x}$ as much as possible unchanged. To ensure that the generated output $\tilde{\mathbf{x}}$ preserves the content of of the original input $\mathbf{x}$ while changing only the conditioning attributes we translate $\tilde{\mathbf{x}}$ back to $\mathbf{x}$. We map $\tilde{\mathbf{x}}$ back to the original input data point, $\mathbf{x}$, by passing $\tilde{\mathbf{x}}$ through the encoder again to get a new latent representation, $\tilde{\mathbf{z}}$, and then passing $\tilde{\mathbf{z}}$ and the original attribute values, $\mathbf{y}$, to the decoder to reconstruct the original input, $\mathbf{x}$. The proposed model is trained jointly end-to-end.

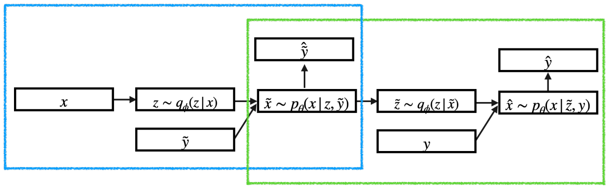

$\hat{\tilde{y}}$
$\hat{y}$
x
$z \sim q_\phi(z|x)$
$\tilde{x} \sim p_\theta(x|z, \tilde{y})$
$\tilde{z} \sim q_\phi(z|\tilde{x})$
$\hat{x} \sim p_\theta(x|\tilde{z}, y)$
$\tilde{y}$
y

Figure 1: Main architecture of the BtVAE model. The model consists of two VAEs with conditional decoders. The first one (blue) modifies the input image conditioning on random sampled attribute values and then the second one (green) translates the modified image back to the original conditioning on the attribute values of the original input.

This procedure is called back translation. The result of back translation is that when we now pass $(\tilde{\mathbf{z}}, \mathbf{y})$ into the decoder, we know how the output should look like, and we can train the model to map $(\tilde{\mathbf{x}}, \mathbf{y})$ into $\mathbf{x}$. The back-translation process is trained by minimizing the back-translation loss, Equation 2 , which is a combination of three terms.

$$\boldsymbol{L}_{bt} = \underbrace{\mathbb{E}_{q_\phi(\tilde{\mathbf{z}}|\tilde{\mathbf{x}})} \log p_\theta(\mathbf{x}|\tilde{\mathbf{z}}, \mathbf{y})}_{A} - \underbrace{D_{\mathrm{KL}}((q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))}_{B} - \underbrace{D_{\mathrm{KL}}(q_\phi(\tilde{\mathbf{z}}|\tilde{\mathbf{x}})\|p(\mathbf{z}))}_{C} \tag{2}$$

The first term is the negative reconstruction cost between the output of the second VAE and the input of the first one. It can be seen as a cycle consistency loss $\mathbb{E}[||\mathbf{x} - \mathrm{Dec}(\mathrm{Enc}(\tilde{\mathbf{x}}), \mathbf{y})||$ that ensures that the content of the input $\mathbf{x}$ will be preserved. Term B penalizes the derivations of the approximate posterior when conditioning on a given input $\mathbf{x}$ from the prior and term C penalizes the derivations of the approximate posterior when conditioning on the modified input $\tilde{\mathbf{x}}$ from the prior.

However, a trivial solution of the model in order to map $(\tilde{\mathbf{x}}, \mathbf{y})$ into $\mathbf{x}$ would be to ignore the conditioning attribute component $\tilde{\mathbf{y}}$ in which case the $\tilde{\mathbf{x}}$ would be simply a reconstruction of the input data $\mathbf{x}$. To overcome this, the generated output $\tilde{\mathbf{x}}$ is passed through the attribute network f_η. In this way we constrain the generated output $\tilde{\mathbf{x}}$ to have the target attributes values, $\tilde{\mathbf{y}}$, i.e. $f(\tilde{\mathbf{x}}) \rightarrow \tilde{\mathbf{y}}$. This is done by minimizing the attribute constraint objective, Equation 3

$$\boldsymbol{L}_{attr} = \boldsymbol{E}_{\tilde{\mathbf{x}} \sim p_\theta(\tilde{\mathbf{x}}|\mathbf{z}, \tilde{\mathbf{y}})}[l(f_\eta(\tilde{\mathbf{x}}), \tilde{\mathbf{y}})] \tag{3}$$

where, η is the prediction model and $l(\cdot)$ represents a loss function. When the attributes are binary labels f_η is an classifier and l is Binary Cross Entropy, while when the attributes are continuous f_η is a regressor and l is a classical mean squared error (MSE).

The final objective of the model is obtained by combining the back-translation objective with the attribute constraint regularizer, Equation 4.

$$\boldsymbol{L} = \boldsymbol{L}_{bt} + \boldsymbol{L}_{attr} \tag{4}$$

At the beginning of the training the only useful information provided to the decoder is the random sampled attribute values $\mathbf{y}$. This, in combination with the attribute network, encourages the decoder to use the provided attributes throughout the entire training process. As the model becomes better trained, the process can be seen as a form of data augmentation, where new inputs with desired attributes $\tilde{\mathbf{x}}, \tilde{\mathbf{y}}$ (which can be out-of-distribution data) are used. By encoding these new inputs $\tilde{\mathbf{x}}$ and decoding them using the original attribute values $\mathbf{y}$, we can manipulate the attributes of the input while maintaining a clear understanding of how the generated output should look like (through the use of back translation). This allows not only us to measure the reconstruction error between the generated output and the desired output, which provides a way to evaluate the model's performance, but also to preserve the content of the original input $\mathbf{x}$.

3 Experiments

3.1 Experimental Setup

Datasets

We evaluate our method using two datasets with independent controlable factors, dSprites (Matthey et al., 2017) and Shapes3D (Burgess & Kim, 2018) and one with continuous attributes, the MNIST (LeCun et al., 2010). Both dSprites (2D shapes) and Shapes3D (3D colored shapes) are generated from 6 ground truth independent latent factors. For the MNIST dataset, instead of conditioning on the digit ID, which is highly informative, we used two continuous attributes, namely stroke width and digit tilt.

To test combinatorial generalisation, following Montero et al. (2021); Montero et al.; Schott et al. (2022), we create disjoint splits of train sets D_{train} and test sets D_{test}. We excluded some combinations of the generative factors in the case of the dSprites and the Shapes3D data and a subset of the MNIST dataset from the training data. We test the capability of out model to reconstruct unseen OOD data, to generate samples conditioning to desired property combinations that they do not exist in the training data and to manipulate the attributes of the OOD data.

To create the training/test split all examples with combinations of a subset of attribute values are excluded from the training set and added to the test set. Thus, an example of a dataset may consist of a training set where all combinations where $[g_1 > 0.5, g_2 > 0.5]$ have been excluded from the training set and have been added to the test set. Note that the model trained on such a datasets would come across a number of examples where $[g_1 > 0.5]$ and also examples where $[g_2 > 0.5]$, but never be trained on an example where both these conditions are true simultaneously. This method was used to create training / test sets for each of the datsets in the following manner:

- dSprites: all images with $xPos > 0.5$ and $yPos > 0.5$ were excluded from the training set. A shape never appear on the right up side of the image, but do appear on the left and down side and right and up side.

- Shapes3D: all images such that $[floorColor < 0.3]$ and $[wallColor > 0.5]$ were removed from the training data. Thus, floor colors in the first third of the HSV spectrum (red, orange.etc) as wall color in the second half of the HSV spectrum (shades of blue, purple, etc) did not appear in the training set. Thus floor colors like red and orange were observed in combination with wall color like red, orange.
- MNIST : all images of the digits 7 and 2 with $-0.9 < StrokeWidth < 1.5$ and $-1 < Tilt < 0.5$ were excluded from the training set. The digit 7 or 2 never appear with StrokeWidth equal to 1. Tilt equal to 0 but the digit 8 could have this property values combination.

BASELINES

BtVAE is based on the framework of the conditional VAE. To assess its effectiveness in addressing OOD reconstruction, attribute manipulation, and conditional generation, we compare it against a conditional VAE as a baseline.. Additionally, we compare our model with the CsVAE (Klys et al., 2018) and the MSP (Li et al., 2020). CsVAE, like BtVAAE is based on a conditional VAE model, but uses two latent variables to separate the information correlated with the attributes $\mathbf{y}$ into a pre-defined subspace. This separation is achieved by minimizing the mutual information $\mathbf{z}$ and $\mathbf{y}$, and results in better control over the generative process. The MSP model, on the other hand, uses orthogonal matrix projection onto subspaces to factor out the information about the attributes of interest $\mathbf{y}$ from the latent variable $\mathbf{z}$. Both the CsVAE and MSP models, similar to our BtVAE model, are supervised, and they both allow for more precise control over the latent representation making possible potential implications in OOD settings.

EVALUATION

As evaluation metric we use the R^2 score based on the MSE score (Schott et al., 2022; Xu et al., 2022). We define the the R^2 score per attribute y_i as

$$R_i^2 = 1 - \frac{MSE_i}{\sigma_i^2} \qquad \text{with} \qquad MSE_j = \mathbb{E}_{(\mathrm{x},\mathrm{y})\sim D_{test}}[(\mathbf{y}_j - f_j(\mathbf{x}))^2] \tag{5}$$

where σ_i^2 is the variance per attribute on the whole dataset. A large R^2 value indicates perfect fit and a value close to zero indicates random guessing since the MSE would be identical to the variance. By utilizing the R^2 score, we can quantitatively evaluate our model's performance in terms of fitting and predicting attributes.

3.2 RESULTS

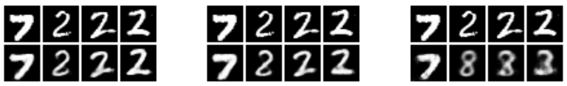
We examine the ability of our model to generate the samples conditioning on property combinations values that have never seen during the training, to reconstruct OOD data and to manipulate the attributes of the OOD data when all the labels are available.

All the models are successful in reconstructing OOD data, as demonstrated in Figures 5, 6. The original images are displayed in the first row of the reconstruction figures, while the corresponding reconstructions appear in the second row. In the case of the Shapes3D dataset, all models show remarkable performance with an R^2 value close to 0.99, as indicated in Table 2. This result highlights the models' ability to effectively leverage the conditioning attributes, floor color and wall color, in the reconstruction process, even in cases where the attribute combinations are not present in the training data. This is further verified visually in Figure 5, where we can see that the models not only maintain the correct colors but also accurately reconstruct the shape.

In the attribute manipulation task, we condition the model on eight different attribute value combinations, with the first four being in-distribution attribute value combinations and the latter four being OOD combinations of attribute values. In Figures 8b, 8c, 8d and 8e, the first four columns display the results of the in-distribution attribute value combinations, while the remaining four columns show the results of the OOD attribute combinations. In both datasets, BtVAE and MSP are able to effectively manipulate the attributes, even for combinations that were not seen during training. cVAE is only able to manipulate the images using in-distribution combinations and struggled to generate data with OOD attribute combinations.

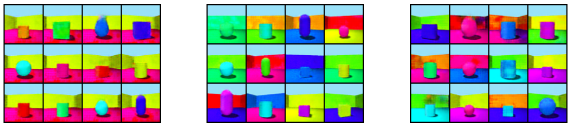
	dSprites		Shapes3d	
Model	Recon.	Manip.	Recon.	Manip.
BtVAE	0.83	0.82	0.99	0.99
MSP	0.76	0.81	0.99	0.99
CsVAE	0.62	0.16	0.99	0.12
cVAE	0.84	0.72	0.99	0.56

2: R^2 score on the reconstruction and attribute manipulation capability of the models using dSprites and Shapes3d datasets



(a) BtVAE **(b)** MSP **(c)** cVAE

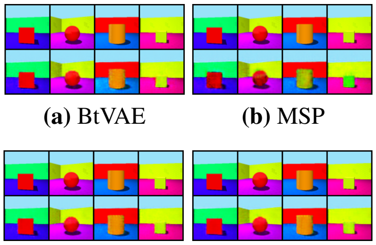
3: MNIST OOD reconstructions



(a) BtVAE **(b)** CsVAE **(c)** CVAE

4: Shapes3D: Samples conditioning on OOD attribute values.

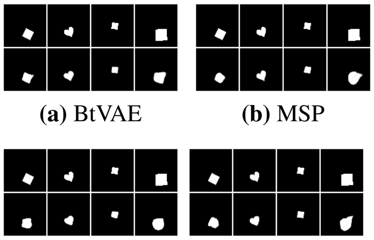
5: Shapes3d OOD reconstructions



(a) BtVAE **(b)** MSP

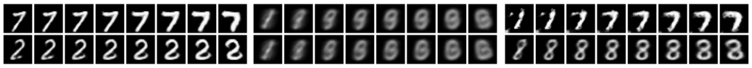
(c) CsVAE **(d)** cVAE

6: dSprited OOD reconstructions



(a) BtVAE **(b)** MSP

(c) CsVAE **(d)** cVAE



(a) BtVAE **(b)** MSP **(c)** cVAE

Figure 7: MNIST, OOD Attribute manipulation

Finally, in the task of conditional generation, BtVAE is the only model capable of generating conditional samples based on OOD attribute values, as shown in Figure 4. By drawing samples from a standard normal prior and conditioning on combinations of attributes that were not present in the training data, the model effectively utilized the target properties while simultaneously generates a diverse range of shapes and coloured objects. This highlights the successful learning of the latent representations and that the model in disentangling the latent representations from the conditioning attributes.

For the MNIST dataset we excluded from the training data a range of values from the stroke width and digit tilt properties of the digits 7 and 2. In the tasks of reconstruction and attribute manipulation, the BtVAE model achieves a R^2 score close to 0.96 in both OOD reconstruction and attribute manipulation, Figures 3a, 7a and it is able to capture the target digit while reconstructing the OOD data. The MSP model is also successful in reconstructing OOD data and manipulating the OOD attribute value combinations, however, it fails in the conditional generation task. On the other hand, the cVAE model struggles in the OOD setting, with a tendency to predict a digit close to the desired one, but with the correct given attribute values, whenever the attribute combination is outside of the training distribution. As we can see in Figures 3c, 7c while the target properties are accurate, but this is not the case for the digit as we can see a digit 2 being replaced with a digit 8.

4 Conclusion

In this paper, we proposed an approach to handle OOD generation, where OOD is defined as combinations of attribute values not observed during training. We evaluated the effectiveness of the BtVAE model in three OOD generation tasks: (1) reconstruction of unseen data, (2) manipulation of attributes using unseen combinations of values, and (3) conditional generation based on unseen attribute value combinations. The results showed that the BtVAE model performs well in all three tasks. Additionally, baseline models also demonstrated promising results in OOD tasks, especially in OOD reconstruction and attribute manipulation, demonstrating that supervised learning can some-

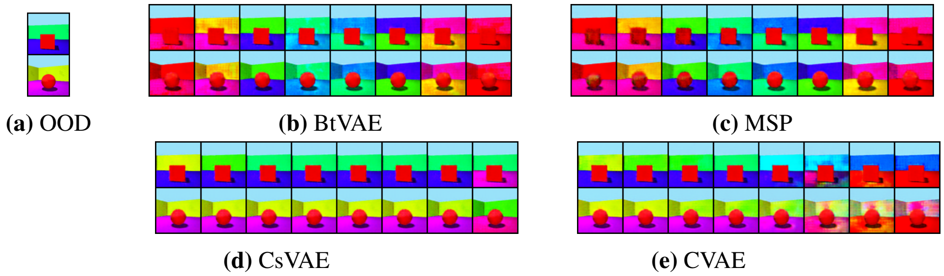


Figure 8: Shapes3D OD Attribute manipulation: using the latent representation of the images in Figure 8a we generate new images by interpolating the floor and wall colour.

times suffice for OOD generation using simple datasets. However, it is crucial to further evaluate the model's performance on more challenging datasets.

References

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Yoshua Bengio, Thomas Mesnard, Asja Fischer, Saizheng Zhang, and Yuhuai Wu. Deep learning of representations: Looking forward. In *International Conference on Learning Representations*, 2018.

Chris Burgess and Hyunjik Kim. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

Ruichu Cai, Zijian Li, Pengfei Wei, Jie Qiao, Kun Zhang, and Zhifeng Hao. Learning disentangled semantic representation for domain adaptation. In *IJCAI: proceedings of the conference*, volume 2019, pp. 2060. NIH Public Access, 2019.

Noam Chomsky. *Aspects of the Theory of Syntax*, volume 11. MIT press, 2014.

Andrea Dittadi, Frederik Träuble, Francesco Locatello, Manuel Wüthrich, Vaibhav Agrawal, Ole Winther, Stefan Bauer, and Bernhard Schölkopf. On the transfer of disentangled representations in realistic settings. In *International Conference on Learning Representations*, 2021.

Irina Higgins, Nicolas Sonnerat, Loic Matthey, Arka Pal, Christopher P Burgess, Matko Bonjak, Murray Shanahan, Matthew Botvinick, Demis Hassabis, and Alexander Lerchner. SCAN: Learning hierarchical compositional visual concepts. In *International Conference on Learning Representations*, 2018.

Ananya Harsh Jha, Saket Anand, Maneesh Singh, and VS Rao Veeravasarapu. Disentangling factors of variation with cycle-consistent variational auto-encoders. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 805–820, 2018.

Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *International conference on machine learning*, pp. 1857–1865. PMLR, 2017.

Jack Klys, Jake Snell, and Richard S. Zemel. Learning latent subspaces in variational autoencoders. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 6445–6455, 2018.

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*, 40, 2017. doi: 10.1017/S0140525X16001837.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018.

Xiao Li, Chenghua Lin, Ruizhe Li, Chaozheng Wang, and Frank Guerin. Latent space factorisation and manipulation via matrix subspace projection. In *International Conference on Machine Learning*, pp. 5916–5926. PMLR, 2020.

Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

Milton L Montero, Jeffrey Bowers, Rui Ponte Costa, Casimir JH Ludwig, and Gaurav Malhotra. Lost in latent space: Examining failures of disentangled models at combinatorial generalisation. In *Advances in Neural Information Processing Systems*.

Milton Llera Montero, Casimir JH Ludwig, Rui Ponte Costa, Gaurav Malhotra, and Jeffrey Bowers. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2021.

L. Schott, J. von Kügelgen, F. Träuble, P. Gehler, C. Russell, M. Bethge, B. Schölkopf, F. Locatello, and W. Brendel. Visual representation learning does not generalize strongly within the same domain. In *10th International Conference on Learning Representations (ICLR)*, April 2022.

Kumar Sricharan and Ashok Srivastava. Building robust classifiers through generation of confident out of distribution examples. *arXiv preprint arXiv:1812.00239*, 2018.

Sachin Vernekar, Ashish Gaurav, Vahdat Abdelzad, Taylor Denouden, Rick Salay, and Krzysztof Czarnecki. Out-of-distribution detection in classifiers via generation. *arXiv preprint arXiv:1910.04241*, 2019.

Nick Watters, Loic Matthey, Chris P. Burgess, and Alexander Lerchner. Spatial broadcast decoder: A simple architecture for disentangled representations in VAEs, 2019.

et al. Wilhelm Von Humboldt, Wilhelm Freiherr von Humboldt. *On Language: On the diversity of human language construction and its influence on the mental development of the human species.* Cambridge University Press, 1999.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6256–6268. Curran Associates, Inc., 2020.

Zhenlin Xu, Marc Niethamme, and Colin Raffel. Compositional generalization in unsupervised compositional representation learning: A study on disentanglement and emergent language. *Conference on Neural Information Processing Systems*, 2022.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.