Generative adversarial interpolative autoencoding: adversarial training on latent space interpolations encourage convex latent distributions

Tim Sainburg
UC San Diego
tsainbur@ucsd.edu

Marvin Thielk
UC San Diego
mthielk@ucsd.edu

Brad Theilman
UC San Diego
btheilma@ucsd.edu

Benjamin Migliori
SPAWAR
migliori@spawar.navy.mil

Timothy Gentner
UC San Diego
tgentner@ucsd.edu

Abstract

We present a neural network architecture based upon the Autoencoder (AE) and Generative Adversarial Network (GAN) that promotes a convex latent distribution by training adversarially on latent space interpolations. By using an AE as both the generator and discriminator of a GAN, we pass a pixel-wise error function across the discriminator, yielding an AE which produces non-blurry samples that match both high- and low-level features of the original images. Interpolations between images in this space remain within the latent-space distribution of real images as trained by the discriminator, and therfore preserve realistic resemblances to the network inputs.

1 Introduction

Generative modeling is quickly becoming an important tool for generating and exploring stimulus spaces in fields such as psychology, linguistics, and neuroscience. The ability to infer abstract and low-dimensional representation's of datasets (as achieved by dimensionality reduction) and to sample from these distributions, to quantitatively vary complex stimulus spaces, provides a powerful tool for psychophysical experimentation.

Two unsupervised neural network based algorithms, the Autoencoder (AE; Hinton and Salakhutdinov 2006) and Generative Adversarial Network (GAN; Goodfellow et al. 2014), are at present the most popular tools in generative modeling. Unsupervised neural network approaches are often ideal generative models because they require little or no tweaking of network architectures to handle entirely new datasets. However successful, a lack of certain constraints on the generative capacity of current neural-network based generative models make it challenging to infer structure from their latent generative representations.

We propose a novel unsupervised generative neural network architecture which controls for the structure of latent representations, and promotes convexity in the model's generative capacity.

1.1 Background on Generative Adversarial Networks (GANs) and Autoencoders (AEs)

An AE is a generative neural network which takes as input X (e.g. a set of images), and is trained to generate a reproduction of the input $G(X)$, by minimizing some error function between the input X and output $G(X)$ (e.g. pixel-wise error). This translation is usually performed after compressing the

Preprint. Work in progress.

representation X into a low-dimensional representation Z, where the number of dimensions in Z is equal to the number of neurons in the compressive layer. The low-dimensional space in which Z lives is called a latent space, and the layer corresponding to Z in the neural network is often called the latent layer. The first half of the network, which translates from X to Z, is called the encoder; the second half of the network, which translates from Z to X, is called the decoder. The combination of these two networks make the AE capable of both dimensionality reduction ($X \rightarrow Z$; Hinton and Salakhutdinov 2006), and the recovery from latent space back into the original high-dimensional space ($Z \rightarrow X$).

GAN architectures comprise two networks, a generator and a discriminator, in which the generator takes as input a latent point, Z, drawn randomly from a distribution (e.g. uniform or normal), and is trained to produce a sample $G(Z)$ in the data domain X. The discriminator takes as input X and $G(Z)$, and is trained to differentiate between real X, and generated $G(Z)$ samples, typically by outputting either a 0 or 1 in a one-neuron layer. The generator is trained to oppose the discriminator by 'tricking' it into categorizing $G(Z)$ samples as X samples. Intuitively, this results in the generator producing $G(Z)$ samples indistinguishable (at least to the discriminator) from those drawn from the distribution X. Thus the discriminator acts as a 'critic' of the samples produced by a generator that is attempting to reproduce the distribution X.

Both GANs and AEs have become very popular in generative modelling, however their generative capacities are fundamentally limited by architectural constraints. Here we describe some of these limitations, and propose a generative architecture, GAIA, that utilizes aspects of both the AE and GAN to negate shortcomings of each architecture independently. Our method provides a novel solution to increasing the stability of network training, the convexity of latent space representations, the preservation of high-dimensional structure in latent space representations, and the bidirectionality of adversarially generated data.

1.2 Convexity of latent space

Generative latent-spaces enable the powerful capacity for smooth interpolations between real-world signals in a high-dimensional space. Linear interpolations in a low-dimensional latent space often produce comprehensible representations when projected back into high-dimensional space (e.g. Engel et al. 2017, Dosovitskiy et al. 2015). In the latent spaces of many network architectures such as AEs, however, linear interpolations are not necessarily justified, because latent-space projections are not trained explicitly to form a convex set.

A convex set of points is defined as a set in which the line connecting any pair of points will fall within the rest of the set. For example, in Figure 1A, the purple distribution represents data projected into a two-dimensional latent space, and the surrounding whitespace represents regions of latent space that do not correspond to the data distribution. This distribution would be non-convex, because a line connecting two points in the distribution (e.g. the black points in Figure 1A) could contain points outside the data distribution (the red region). In an AE, if the red region of the interpolation were sampled, projections back into the high-dimensional space may not necessarily correspond to realistic exemplars of X, because that region of Z does not belong to the true data distribution.

One approach to overcoming non-convexity in a latent space is to force the latent representation of the dataset into a pre-defined distribution (e.g. a normal distribution), as is performed by Variational Autoencoders (VAEs; Kingma and Welling 2013). By constraining the latent space of a VAE to fit a normal distribution, the latent space representations are encouraged to belong to a convex set. This method, however, pre-imposes a distribution in latent space that may be a suboptimal representation of the high-dimensional dataset. Standard GAN latent distributions are sampled directly, similarly allowing arbitrary convex distributions to be explicitly chosen for latent spaces. In both cases, hard-coding the distributional structure of the latent space may not respect the high-dimensional structure of the original distribution.

1.3 Pixel-wise error and bidirectionality

AEs tend to produce blurry images due to their pixel-wise error functions [Larsen et al., 2015], which minimize error by smoothing the sharp contrasts (edges) present in real images. GANs do not suffer from this blurring problem, becasue they are not trained to reproduce input data, but rather to generate data that could plausibly belong to the true distribution X. Because the discriminator is trained to

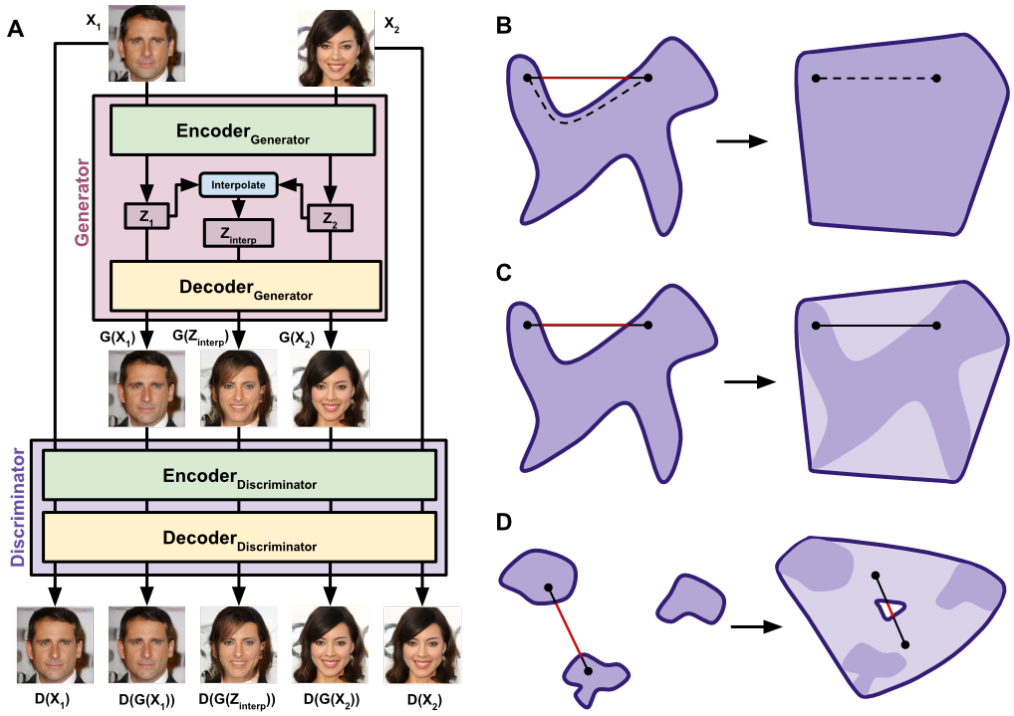

Figure 1: (A) GAIA network architecture. (B) An example AE latent distribution (purple) which is non-convex. Black circles represent two projections from X to Z and the solid line between them represents an interpolation, in which the red portion are points in the interpolation that do not correspond to the distribution of X. The dashed line represents an interpolation which passes through only the true data distribution. One hypothesis is that the distribution on the left will warp to produce linearly interpolations within the distribution. (C) An alternative hypothesis is that by training the network upon interpolations, samples in the interpolated regions (lighter purple) will be trained to generate data similar to the X, without manipulating the distribution of Z. (D) The distribution remains non-convex, because two point interpolations do not reach all of the spaces in between the interpolated data distribution.

recognize image features, such as blurriness, that could potentially distinguish between X and $G(X)$, the discriminator of GANs enforces the generation of images that are not blurry. Producing data that fits into the distribution of X, rather than reproducing individual instances of X_i comes at a cost however. While AEs learn both the translation from X to Z and Z to X, GANs only learn the latter ($Z \rightarrow X$). In other words, the pixel-wise error function of the AE produces blurry images but is bidirectional, while the discriminator-based error function of the GAN produces sharp images and is unidirectional.

2 Generative Adversarial Interpolative Autoencoding (GAIA)

Our model, GAIA (Figure 1 left), acts as a GAN in which both the generator and the discriminator are AEs. The discriminator is trained to minimize the pixel-wise error (ℓ_1) between input real images (X) and their AE reproduction in the discriminator ($D(X)$) while maximizing the AE error between images generated ($G(X)$) by the generator and their reproduction in the discriminator ($D(G(X))$):

$$\|X - D(X)\|_1 - \|G(X) - D(G(X))\|_1$$

The generator is trained on the inverse, to minimize the pixel-wise error between input (X) and output ($D(G(X))$) images such that the discriminator reproduces the generated images to be as close to the original image as possible:

$$\|G(X) - D(G(X))\|_1$$

Figure 2: Interpolations between autoencoded images in our network. The farthest left and right columns correspond to the input images, and the middle columns correspond to a linear interpolation in latent-space.

Using an AE as a generator has been previously been used in the VAE-GAN [Larsen et al., 2015], and decreases blurring from pixel-wise error in AEs at the expense of exact-reproduction of images. Similarly, using a discriminator as an AE has been previously used in BEGAN [Berthelot et al., 2017], which has been shown to improve stability in GANs but remains unidirectional[1]. In GAIA, we combine these two architectures, allowing the generator to be trained on a pixel-wise error that is passed across the discriminator, explicitly reproducing images as in an AE, while producing sharper images characteristic of a GAN.

In addition, linear interpolations are taken between the latent-space representations of images:

$$Z_{interp} = interpolate(Z^1_{gen}, Z^2_{gen})$$

Where interpolations are Euclidean interpolations between pairs of points in Z, sampled from a Gaussian distribution around the midpoint between Z^1_{gen} and Z^2_{gen}. The midpoints are then passed through the decoder of the discriminator, which are treated as generated samples by the GAN error function:

$$\|G(Z_{interp}) - D(G(Z_{interp}))\|_1$$

The discriminator is trained to maximize this error, and the generator is trained to minimize this error.

In full, the loss of the discriminator, as in BEGAN, is to minimize pixel-wise error of real data, and maximize pixel-wise error of generated data (including interpolations):

$$\begin{aligned} \mathcal{L}_{Disc} = &\|X - D(X)\|_1 + \\ &\|G(X) - D(G(X))\|_1 + \\ &\|G(Z_{interp}) - D(G(Z_{interp}))\|_1 \end{aligned} \tag{1}$$

The loss of the generator is to minimize the error across the descriminator for the input images inputted into the generator ($D(G(X))$), along with the minimizing the error of the interpolations generated by the generator ($D(G(Z_{interp}))$).

$$\begin{aligned} \mathcal{L}_{Gen} = &\|X - D(G(X))\|_1 - \\ &\|G(Z_{interp}) - D(G(Z_{interp}))\|_1 \end{aligned} \tag{2}$$

In summary, the generator and discriminator are both AEs, allowing us to pass a pixel-wise error across the two AEs, allowing the generator to produce images which are autoencoded but not blurry. We also train the network on interpolations in the generator, to explicitly train the generator to produce interpolations which deceive the discriminator. As a result, our inference method more greatly resembles the original data than other GAN-based methods (Figure 5).

[1] Although it is possible to find the regions of Z most closely corresponding to X

2.1 Preservation of local-structure in high-dimensional data

VAEs and GANs force the latent distribution of a dataset of faces to into a pre-defined distribution, for example the Gaussian or uniform distributions. This approach presents a number of advantages, such as ensuring latent space convexity and thus being better able to sample from the distribution. However, these benefits are gained at the loss of respecting the structure distribution of the original dataset in high-dimensional space. To better respect the original high-dimensional structure of the dataset, we impose a regularization between the latent space representations of the data (Z), and the original high dimensional dataset (X).

For each minibatch presented to the network, we compute a loss for the distance between the log of the pairwise Euclidean distances of samples in X and Z space:

$$\mathcal{L}_{dist}(X,Z) = \frac{1}{B}\sum_{i,j}^{B}\left[log_2\left(1+\frac{(X_i-X_j)^2}{\frac{1}{N}\sum_{i,j}(X_i-X_j)^2}\right) - log_2\left(1+\frac{(Z_i-Z_j)^2}{\frac{1}{N}\sum_{i,j}(Z_i-Z_j)^2}\right)\right]^2$$

We then apply this error term to the generator to encourage the pairwise distances of the minibatch in latent space to be similar to the pairwise distances of the minibatch in the original high-dimensional space. A hyperparameter (α) is used to balance the emphasis of the structure loss against the loss of the reconstruction/adversarial losses, which was set at half the learning rate ($1^{-4}/2$).

2.2 Instability in adversarial networks

GANs are notoriously challenging to train, and refining techniques to balance and properly train GANs has been an area of active research since the conception of the GAN architecture (e.g. Berthelot et al. 2017, Salimans et al. 2016, Mescheder et al. 2017, Arjovsky et al. 2017). In traditional GANs, a balance needs to be found between training the generator and discriminator, otherwise one network will overpower the other and the generator will not learn a representation which fits the dataset. With GAIA, additional balances are required, such as between reproducing real images vs. discriminating against generated images, or balancing the generator of the network toward emphasizing autoencoding vs. producing high-quality latent-space interpolations.

We propose a novel, but simple, GAN balancing act which we find to be very effective. In our network, we balance the GAN's loss using a sigmoid centered at zero:

$$sigmoid(d) = \frac{1}{1+e^{-d*b}}$$

In which b is a hyper-parameter representing the slope of the sigmoid[2], and d is the difference between the two values being balanced in the network. For example, the balance in the learning rate of the discriminator and generator is based upon the loss of the real and generated images:

$$\delta_{Disc} \leftarrow sigmoid(\|X - D(X)\|_1 - \|X - D(G(X))\|_1 + \|G(Z_{interp}) - D(G(Z_{interp}))\|_1/2)$$

The learning rate of the generator is then set as the inverse:

$$\delta_{Gen} \leftarrow 1 - \delta_{Disc}$$

This allows each network to catch up to the other network when it is performing worse. The same principles are then used to balance the different losses within the generator and discriminator, which can be found in Algorithm 1. This balancing act allows the part of the network performing more poorly to be emphasized in the training regimen, resulting in more balanced and stable training.

2.3 Network Architecture

In principle, any form of AE network can be used in GAIA. In the experiments shown in this paper, we used a modified version of network architecture advocated by Huang et al. [2018], which is comprised of a style and content AE using residual convolutional layers. Each layer of the decoder

[2]kept at 20 for our networks

Figure 3: Attribute vectors added to Z representation of different images from the CELEBA-HQ dataset.

uses half as many filters as the encoder, and a linear latent layer is used in the encoder network but not the decoder network. The final number of latent neurons for the style and content networks are both 512 in the 128×128 pixel model shown here. A Python/Tensorflow implementation of this network architecture is linked in the Conclusions section, and more details about the network architecture used are located in Huang et al. [2018].

2.4 Feature manipulation

We find that, like several other generative models, high-level features correspond to linear vectors in GAIA's latent space. High level feature representations are typically determined using the means of Z representations of images containing features (e.g. Radford et al. 2015, Larsen et al. 2015, Kingma and Dhariwal 2018). The mean of the latent representations of the faces in the dataset (here CELEBA-HQ) containing an attribute (Z_{feat}) and not containing that attribute (Z_{nofeat}) are subtracted ($Z_{feat} - Z_{nofeat}$) to acquire a high level feature vector. The feature vector is then added to, or subtracted from, the latent representation of individual faces (Z_i), which is passed through the decoder of the generator, producing an image containing that high-level feature. We find, however, that these features are often tangled together in the CELEBA-HQ dataset. For example, adding a latent vector to make images look *older* biases the image toward *male*, and making the image look more *young* biased the image toward *female*. This likely happens because the ratio of young males to older males is 0.28:1, whereas the ratio of young females to older females is much greater at 8.49:1 in the CELEBA-HQ dataset. To overcome this, we use an ordinary least-squares regression trained to predict Z representations from feature attributes and use the coefficients of the model as feature vectors. We find that these features are less intertwined than subtracting means alone, which can be seen in Figure 4, where attribute vectors alone are passed through the decoder network.

2.5 Related work

This work builds primarily upon the GAN and AE. We used the AE as a discriminator motivated by Berthelot et al. [2017], and an AE as a generator motivated by Larsen et al. [2015]. As noted above, a number of other adversarial network architectures for inference have been designed in the past several years. We compare our inference method to other GAN based methods in Figure 5. Similar motivations for better bidirectional inference-based methods have also been explored using flow-based generative models [Kingma and Dhariwal, 2018, Dinh et al., 2014], which do not rely on

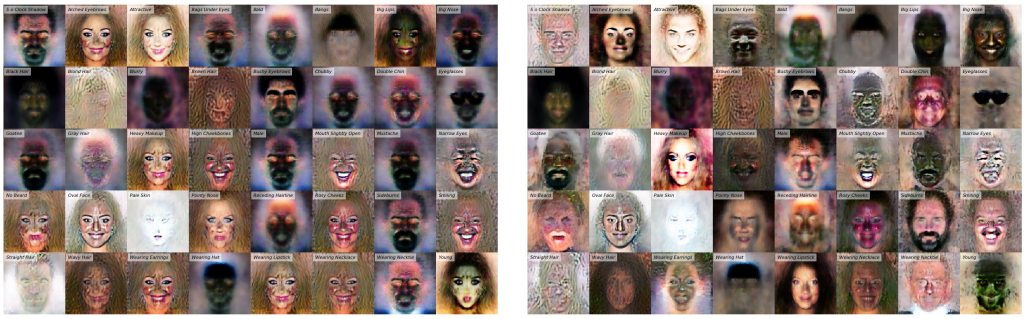

Figure 4: Attribute vectors passed through the decode of the network. Attributes to the left are found by subtracting means. Attributes to the right are found using ordinary least-squares regression. The attribute vectors to the right are more variable. Zoom-in to see labels.

Figure 5: We compare the output of our generator to other inference and autoencoding based algorithms with adversarial loss. Input images (X), and network reconstruction images ($G(X)$) are shown side by side, with inputs on the left. (A) Larsen et al. [2015] (B) Mescheder et al. [2017] (C) Berthelot et al. [2017] (D) Dumoulin et al. [2016] (E) Ulyanov et al. [2017] F Our method.

adversarial loss. Due to their exact latent-variable inference, these architectures may also provide a useful direction for developing generative models to explore latent-spaces of data for generating datasets for psychophysical experiments.

3 Conclusion

We propose a novel architecture of GAN in which both the generator and discriminator are AEs. In this architecture, a pixel-wise loss can be passed across the discriminator producing autoencoded images without blurry reconstructions. Further, using the adversarial loss of the GAN, we train the generator's AE explicitly on interpolations between samples projected into latent space, promoting a convex latent space representation. This method more explicitly lends itself to interpolations between complex signals using a neural network latent space, while still respecting the high-dimensional structure of the input data.

The proposed architecture still leaves much to be accomplished, and modifications of this architecture may prove to be more useful, for example utilizing different encoder strategies such as progressively growing layers [Karras et al., 2017], interpolating across the entire minibatch rather than two-point interpolations, or finding other methods to train more explicitly on a convex space using adversarial

loss. Further explorations are also needed to understand how interpolative sampling effects the structure of the latent space of GAIA.

Our network architecture furthers generative modeling by providing a novel solution to maintaining pixel-wise reconstruction over an adversarial architecture. Further, we take a step in the direction of convex latent space representations in a generative context. This architecture should prove useful both for current behavioral scientists interested in sampling from smooth and plausible stimuli spaces (e.g. Zuidema et al. 2018), as well as providing motivation for future solutions to bridging the gap between data and the generative models used capture it.

Our network was trained using Tensorflow, and our full model, code, and high resolution images are available on GitHub. Videos of a trained network interpolating between real images and adding feature vectors are available at the blog post associated with this paper.

4 Acknowledgements

Work supported by NSF Graduate Research Fellowship 2017216247 to TS.

Bibliography

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

David Berthelot, Thomas Schumm, and Luke Metz. Began: boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. Learning to generate chairs with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1538–1546, 2015.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi. Neural audio synthesis of musical notes with wavenet autoencoders. *arXiv preprint arXiv:1704.01279*, 2017.

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*, pages 2672–2680, 2014.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. *arXiv preprint arXiv:1804.04732*, 2018.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

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

Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015.

Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*, 2017.

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

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Adversarial generator-encoder networks. *CoRR, abs/1704.02304*, 2017.

Willem Zuidema, Robert M. French, Raquel G. Alhama, Kevin Ellis, Tim O'Donnell, Tim Sainburg, and Timothy Q. Gentner. Five ways in which computational models can help advancing artificial grammar learning research. *in press*, 2018.

Algorithm 1 Training the GAIA Model

$\theta_{Gen}, \theta_{Disc} \leftarrow$ initialize network parameters
repeat
    $X \leftarrow$ random mini-batch from dataset
    # pass through network
    $Z \leftarrow G_{Enc}(X)$
    $Z_{interp} \leftarrow interpolate(Z)$
    $X_{gen} \leftarrow G_{Dec}(Z)$
    $X_{interp} \leftarrow G_{Dec}(Z_{interp})$
    $\tilde{X} \leftarrow D(X)$
    $\tilde{X}_{interp} \leftarrow D(X_{interp})$
    $\tilde{X}_{gen} \leftarrow D(X_{gen})$
    # compute losses
    $\mathcal{L}_X \leftarrow \|X - \tilde{X}\|_1$
    $\mathcal{L}_{X_{gen}} \leftarrow \|X - \tilde{X}_{gen}\|_1$
    $\mathcal{L}_{X_{interp}} \leftarrow \|X_{interp} - \tilde{X}_{interp}\|_1$
    $\mathcal{L}_{distance} \leftarrow pairwisedistance(X, Z_{gen})$
    # balance losses
    $\delta_{Disc} \leftarrow sigmoid(\mathcal{L}_X - mean(\mathcal{L}_{X_{gen}}, \mathcal{L}_{X_{interp}}))$
    $\delta_{Gen} \leftarrow 1 - \delta_{Disc}$
    $\mathcal{W}_{Gen_{interp}} \leftarrow sigmoid(\mathcal{L}_{X_{interp}} - \mathcal{L}_{X_{gen}})$
    $\mathcal{W}_{Gen_{gen}} \leftarrow 1 - \mathcal{W}_{Gen_{interp}}$
    $\mathcal{W}_{Disc_{fake}} \leftarrow sigmoid(mean(\mathcal{L}_{X_{gen}}, \mathcal{L}_{X_{interp}}) \cdot \gamma - \mathcal{L}_X)$
    # update parameters according to gradients
    $\theta_{Gen} \overset{+}{\leftarrow} -\Delta_{\theta_{Gen}}(\mathcal{L}_{X_{gen}} \cdot \mathcal{W}_{Gen_{gen}} + \mathcal{L}_{X_{interp}} \cdot \mathcal{W}_{Gen_{interp}} + \mathcal{L}_{distance} * \alpha) \cdot \delta_{Disc}$
    $\theta_{Disc} \overset{+}{\leftarrow} -\Delta_{\theta_{Disc}}(\mathcal{L}_X - mean(\mathcal{L}_{X_{gen}}, \mathcal{L}_{X_{interp}}) \cdot \mathcal{W}_{Disc_{fake}}) \cdot \delta_{Gen}$
until deadline