NEURAL PHOTO EDITING WITH INTROSPECTIVE ADVERSARIAL NETWORKS

Andrew Brock, Theodore Lim,& J.M. Ritchie
School of Engineering and Physical Sciences
Heriot-Watt University
Edinburgh, UK
{ajb5, t.lim, j.m.ritchie}@hw.ac.uk

Nick Weston
Renishaw plc
Research Ave, North
Edinburgh, UK
Nick.Weston@renishaw.com

ABSTRACT

We present the Neural Photo Editor, an interface for exploring the latent space of generative image models and making large, semantically coherent changes to existing images. Our interface is powered by the Introspective Adversarial Network, a hybridization of the Generative Adversarial Network and the Variational Autoencoder designed for use in the editor. Our model makes use of a novel computational block based on dilated convolutions, and Orthogonal Regularization, a novel weight regularization method. We validate our model on CelebA, SVHN, and ImageNet, and produce samples and reconstructions with high visual fidelity.

1 INTRODUCTION

Recent advances in generative models for images have enabled the training of neural networks that produce image samples and interpolations with high visual fidelity. Two key methods, the Variational Autoencoder (VAE)(Kingma & Welling, 2014) and Generative Adversarial Network (GAN)(Goodfellow et al., 2014), have shown great promise for use in modeling the complex, high-dimensional distributions of natural images. VAEs are probabilistic graphical models that learn to maximize a variational lower bound on the likelihood of the data by projecting into a learned latent space, then reconstructing samples from that space. GANs learn a generative model by training one network, the "discriminator," to distinguish between real and generated data, while simultaneously training a second network, the "generator," to produce samples which the discriminator cannot distinguish from real data. Both approaches can be used to generate images by sampling in a low-dimensional learned latent space, but each comes with its own set of benefits and drawbacks.

VAEs have stable training dynamics, but when trained using elementwise L2 distance as a reconstruction objective produce blurry images, as the learned model is conservative and tends to "hedge its bets." Using the intermediate activations of a pre-trained discriminative neural network as features for comparing reconstructions to originals (Lamb et al., 2016) mollifies this effect, but requires labels in order to train the discriminative network in a supervised fashion.

By contrast, GANs have unstable and often oscillatory training dynamics, but produce images with sharper, more photorealistic features. Basic GANs lack an inference mechanism, though techniques to adversarially train an inference network (Dumoulin et al., 2016) (Donahue et al., 2016) have recently been developed, as well as a hybridization that uses the VAE's inference network (Larsen et al., 2015).

Standard procedure for evaluating these models involves generating random samples or reconstructions, and interpolating between generated images. Achieving a specific change in the model output, such as changing an unsmiling face to a smiling face, usually requires that the learned latent space be augmented during training with a set of labeled attributes, such that interpolating along the model's latent "smile vector" produces a specific change. In the fully unsupervised setting, there is no guarantee that a particular latent variable will directly control a semantically meaninigful output feature.

In this paper, we present the Neural Photo Editor, a novel interface for exploring the latent space of generative models. Our method makes it possible to produce specific semantic changes in the output image by use of a "contextual paintbrush" that indirectly modifies the latent vector. By applying

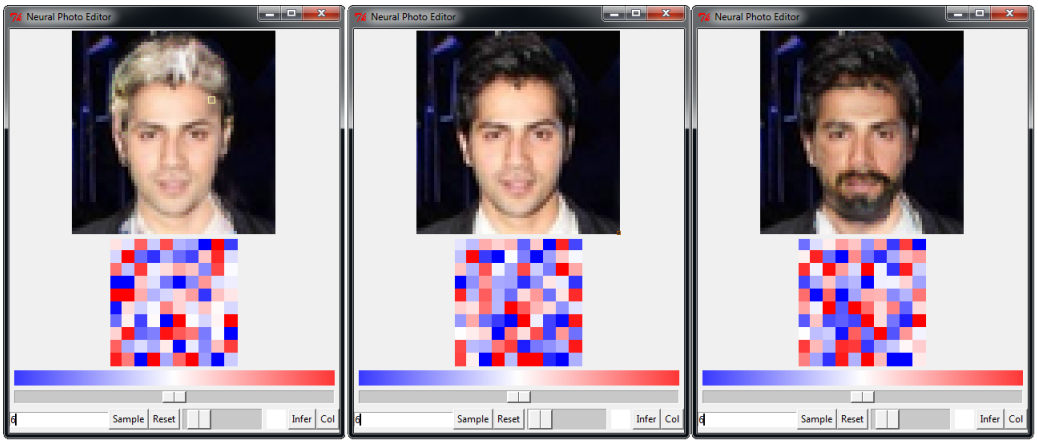

Figure 1: The Neural Photo Editor. The original image is center. The red and blue tiles are visualizations of the latent space, and can be directly manipulated as well.

a simple interpolating mask, we enable this same exploration for existing photos, even with an imperfect reconstruction of that photo.

Complementary to the Neural Photo Editor, we present the Introspective Adversarial Network (IAN), a novel hybridization of the VAE and the GAN that leverages the power of the adversarial objective while maintaining the efficient inference mechanism of the VAE. Our model makes use of a novel inception-style convolutional block based on dilated convolutions (Yu & Koltun, 2016) and Orthogonal Regularization, a novel technique for regularizing weights in convolutional neural networks. Qualitative experiments on CelebA (Liu et al., 2015), SVHN (Netzer et al., 2011) and Imagenet(Russakovsky et al., 2015) demonstrate the sampling, reconstructing, and interpolating capabilities of the IAN, whille competitive performance on the semi-supervised SVHN classification task quantitatively demonstrates its inference capabilities.

2 Neural Photo Editing

Standard methods for exploring the latent space of a generative model involve interpolating between two samples or directly manipulating the latent space. Directly manipulating latent variables to produce a specific change works well when the network is provided descriptive labels, but is far less effective when the network is trained in a wholly unsupervised fashion, as there is no guarantee that individual latent vectors will correspond to semantically meaningful features.

We present an interface,shown in Figure 1, that allows for a more intuitive exploration of a generative model by indirectly manipulating the latent space with a "contextual paintbrush." The key idea is simple: a user selects a paint brush size and color (as with a typical image editor) and paints on the output image. Instead of changing individual pixels, the interface backpropagates the difference between the local image patch and the requested color, and takes a gradient descent step in the latent space to minimize that difference. This step results in globally coherent changes that are semantically meaningful in the context of the requested color change.

For example, if a user has an image of a person with light skin, dark hair, and a widow's peak, by painting a dark color on the forehead, the system will automatically add hair in the requested area. Similarly, if a user has a photo of a person with a closed-mouth smile, the user can produce a toothy grin by painting bright white over the target's mouth. This method is non-iterative in the sense that a single gradient descent step is taken every time the user requests a change, and runs smoothly in real-time on a modest laptop GPU.

This method works well for samples directly generated by the network, but fails when applied directly to existing photos, as it relies on the manipulated image being directly controlled by the latent variables. Reconstructing images that have passed through such a representational bottleneck (i.e.

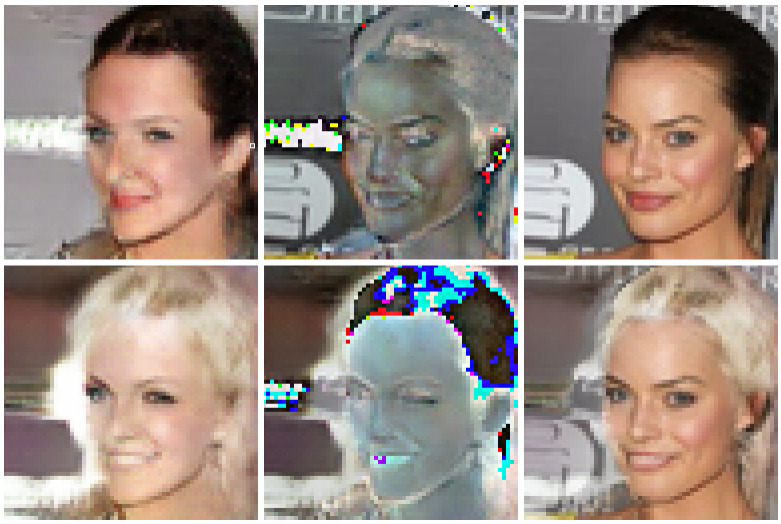

Figure 2: Visualizing the interpolation mask. Top, left to right: Reconstruction, reconstruction error, original image. Bottom: Modified reconstruction, Δ, output.

with an autoencoder) is difficult, and certain to produce reconstructions which, lacking pixel-perfect accuracy, are useless for making small changes to natural images.

To combat this, we introduce a simple masking technique that allows a user to edit images by modifying a given photo's reconstruction, then transferring those changes to the original image. We take the output image to be a sum of the reconstruction, and a masked combination of the requested pixel-wise changes and the reconstruction error:

$$Y = \hat{X} + M\Delta + (1 - M)(X - \hat{X})$$

Where X is the original image, $\hat{X}$ is the model's reconstruction, and Δ is the difference between the modified reconstruction and $\hat{X}$. The mask M is the channel-wise mean of the absolute value of Δ,typically smoothed with a Gaussian filter and truncated to be between 0 and 1:

$$M = min(g(|\bar{\Delta}|), 1)$$

A visualization of the masking technique is shown in Figure 2. This method adds minimal computational cost to the underlying latent space exploration and produces convincing changes of features including hair color and style, skin tone, the presence or absence of glasses, and facial expression. A video of the interface in action is available online.[1]

3 Introspective Adversarial Networks

Complementary to the Neural Photo Editor, we introduce the Introspective Adversarial Network (IAN), a novel hybridization of the VAE and GAN. Similar to VAE/GAN (Larsen et al., 2015), we use the decoder network of the autoencoder as the generator network of the GAN, but instead of training a separate discriminator network, we combine the encoder and discriminator into a single network. We train the network to simultaneously generate photorealistic samples, infer latent values directly from an image, and reconstruct an image from inferred latents. We use three distinct loss functions:

- L1 pixel-wise reconstruction loss, which we prefer to the L2 reconstruction loss for its higher average gradient.

[1] https://www.youtube.com/watch?v=FDELBFSeqQs

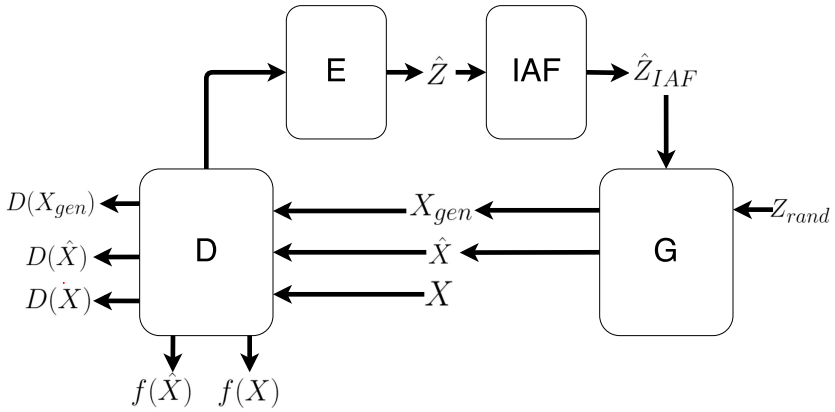

Figure 3: The Introspective Adversarial Network (IAN).

- Feature-wise reconstruction loss, evaluated as the L2 difference between the original and reconstruction in the space of the hidden layers of the discriminator.
- Ternary Adversarial Loss, a modification of the Adversarial loss that forces the network to label a sample as real, generated, or reconstructed (as opposed to a binary real vs. generated label).

Central to the IAN is the idea that the discriminator learns a hierarchy of features that are useful for multiple tasks, including inferring latents and comparing samples. The discriminator itself is updated solely using the ternary adversarial loss, and the network instead learns an encoding subnetwork, implemented as a fully-connected layer on top of the final convolutional layer of the discriminator. The generator is thus trained to produce random images (from Z_{rand}) which are indistinguishable from real images X, as well as reconstructions $\hat{X}$ (from $\hat{Z}_{IAF}$) which are simultaneously photorealistic, and similar to the original image in both the pixel feature spaces. ate activations of the discriminator, as well as being photorealistic. The IAN architecture is depicted in Figure 3.

The loss functions for each network thus become:

$$\mathcal{L}_G = log(D(X|G(Z_{rand}))) + log(D(X|\hat{X})) + ||X - \hat{X}||_1 + ||f(X) - f(\hat{X})||_2$$

$$\mathcal{L}_D = (1 - log(D(G(Z_{rand}|G(Z_{rand}))))) + (1 - log(D(\hat{X}|\hat{X}))) + log(D(X|X))$$

3.1 Feature-wise and Ternary Adversarial Loss

Comparing the outputs of intermediate encoder/discriminator layers was originally inspired by Discriminative Regularization (Lamb et al., 2016), though we note that Feature Matching (Salimans et al., 2016) is designed to operate in a similar fashion, but without the guidance of an inference mechanism to match latent values Z to particular values of $f(G(Z))$. We find that using this loss to complement the pixel-wise difference results in sharper reconstructions that are semantically similar (i.e. toothy grins stay toothy, brown hair stays brown).

We note that it is possible for the discriminator to get an early lead on the generator and slow training, perhaps because it learns a small subset of features (artifacts in the generator's output) that distinguish real and generated samples, reducing the range of features the generator can learn from it. The ternary adversarial loss, where the discriminator attempts to assign one of three labels to a sample, presents a more difficult task for the discriminator, and reduces the likelihood of it learning such a small feature space by forcing it to distinguish between reconstructions and random samples. We posit that this also leads to the discriminator ultimately learning a richer feature space, contributing to consistent sample quality.

We also experiment with using the style loss of neural style transfer (Gatys et al., 2015) for reconstruction comparison. We find that the style loss improves early training but comes at a prohibitive

memory cost. We also experimented with evaluating the feature-wise reconstruction loss using the output of each layer before its nonlinearity (in the style of Pre-activation (He et al., 2016)), but found this to significantly decrease performance.

3.2 IAF WITH RANDOMIZED MADE

We take advantage of the auto-encoding nature of our architecture and implement the MADE (Germain et al., 2015) variant of Inverse Autoregressive Flow (IAF) (Kingma et al., 2016). In initial experiments, we found that implementing full MADE (including shuffling the ordering and connectivity masks) significantly reduced results quality, perhaps because the shuffling injected an undesirable amount of internal covariate shift. We found that using only a single initial shuffle and orthogonally initializing (Saxe et al., 2014) but not training the MADE worked best, suggesting that IAF whitening can be performed using any random autoregressive function of the latents.

3.3 ARCHITECTURE

Our model has the same basic structure of the DCGAN (Radford et al., 2015), augmented with Multiscale Dilated Convolution (MDC) blocks between successive upsampling layers in the generator and an autoregressive RGB block at the output of the generator and Minibatch Discrimination (Salimans et al., 2016) in the discriminator. We found that using Batch Normalization (Ioffe & Szegedy) and Adam (Kingma & Ba, 2014) were essential to successfully training IANs. Code containing all experiments and exact architectural details is available online.[2]

We found that when designing IANs, maintaining the "balance of power" between the generator and the discriminator to be key. In particular, we found that if we made the discriminator too expressive (i.e. by inserting MDC blocks in between downsampling layers) it would quickly out-learn the generator and achieve near-perfect accuracy, resulting in a significant slow-down in training. We thus maintain an "improvement ratio" rule of thumb, where every layer we add to the discriminator was accompanied by an addition of three layers in the generator.

We put an especial focus on representational bottlenecks in our design. Just as aggressive downsampling in the early layers of a classification network can hamper performance, we posit that aggressive upsampling in the final layers of the generator can reduce performance by hampering the backpropagation of error. This supposition guides us to add more expressivity (by means of additional MDC blocks) in the later layers of the generator compared to the early layers.

3.3.1 MULTISCALE DILATED CONVOLUTION BLOCKS

We propose a novel Inception-style (Szegedy et al., 2016) convolutional block motivated by the ideas that image features naturally occur at multiple scales, and that a network's expressivity is proportional to the range of functions it can represent divided by its total number of parameters. The Multiscale Dilated Convolution (MDC) block applies a single 3x3 filter at multiple dilation factors, then performs a weighted elementwise sum of each dilated filter's output.

As shown in Figure 4, each block is thus parameterized by a bank of N 3x3 filters, applied with S different factors of dilation, a 1x1 convolution with weights taken as the mean of the main 3x3 filter, and a set of N*(S+1) weights k, which relatively weight the output of each filter at each scale.

Each block has a total of N*(9*C+S+1) weights, N*(S+1) more than a single 3x3 convolution. For a typical hidden conv layer of 512 filters with a 256 channel input, a 3x3 convolutional block has 1.12M parameters, and an equivalent MDC block applied at 3 scales adds a mere 1500 parameters, less than 1%.

We performed initial explorations by making use of MDC blocks in a 10-layer Residual network architecture for CIFAR-100 (Krizhevsky & Hinton, 2009), and found that the network converged to 71% test accuracy within just 10 epochs. Though this is 10 percent lower than the current state-of-the-art, we noted the high performance-to-depth ratio and the fast training time, and immediately integrated the blocks into our IAN architecture, leaving full discriminative validation to future work.

[2] https://github.com/ajbrock/Neural-Photo-Editor

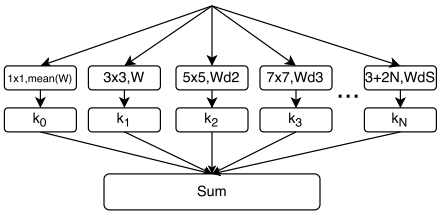

Figure 4: Multiscale Dilated Convolution Block. Each block is completely parameterized by a bank of N 3x3 filters, W, the number of scales it operates at, S, and a set of S+1 N-dimensional coefficients, k, that weight the relative output of each filter at each scale.

3.3.2 RGB MODELING

Typically, the output layer of an image-generating neural network consists of 3 HxW channels corresponding to the R-G-B color channels, which are each fed to a squashing nonlinearity and mapped to an 8-bit color scheme.

PixelRNNs(van den Oord et al., 2016) changed this by allowing the network to specify a 256-dimensional discrete distribution, with each dimension corresponding to a unique unsigned 8-bit value, reasoning that allowing the network to specify a flexible discrete color distribution would improve image quality. We adopt a similar view, but note that outputting a 3*H*W*256 dimensional output from a convolutional layer is computationally expensive. Instead, we reason that we need not specify a full discrete distribution, but can instead output the shape parameters of a flexible continuous PDF, such as a beta distribution, which then allows us to impose priors on the desired shape of that distribution, rather than having the network concentrate all of its probability mass on a single point. In practice, we find that training the network to specify a beta distribution rather than a single point works well even without enforcing a prior on the shape of the distribution, reducing the likelihood of "washed-out" samples and improving the fidelity of reconstructions.

As in PixelRNNs, we autoregressively specify the color channels, meaning that the R channel is dependent on the output of the last hidden layer, the G channel is dependent on the last hidden layer and the R channel, and the B channel is dependent on the last hidden layer and both the R and G channels.

3.4 ORTHOGONAL REGULARIZATION

Orthogonality is a desirable quality in neural net matrices, partially because multiplication by an orthogonal matrix leaves the norm of the original matrix unchanged. This property is valuable in deep or recurrent networks, where repeated matrix multiplication can result in signals vanishing or exploding. We note the success of initializing weights with orthogonal matrices (Saxe et al., 2014), and posit that maintaining orthogonality throughout training is also desirable. To this end, we propose a simple weight regularization technique, Orthogonal Regularization, that encourages weights to be orthogonal by pushing them towards the nearest orthogonal manifold. We add a cost term to our objective function:

$$\mathcal{L}_{ortho} = \Sigma(WW^T - I)$$

Where W is a convolutional kernel and I is the identity matrix. As the orthogonality condition is underconstrained for matrices with spatial extent > 1, there are an infinite number of possible orthogonal matrices, meaning this technique does not constrain the learned features to a limiting subspace, and can be used in conjunction with other regularizers such as L2.

We found that applying Orthogonal Regularization to our CIFAR-100 testbed immediately improved accuracy by 2% without any other changes, and we thus make use of it in all of our experiments.

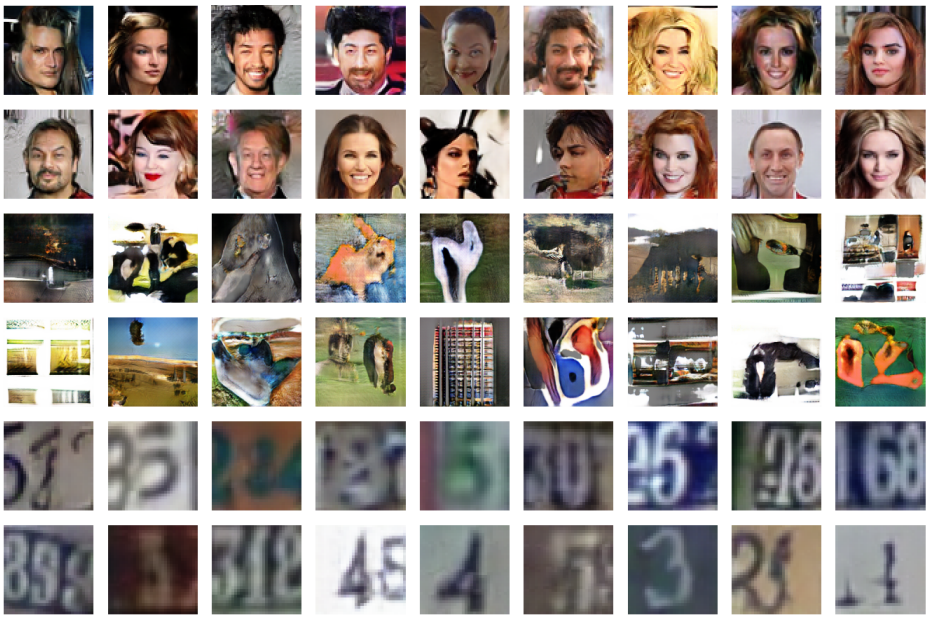

Figure 5: CelebA, ImageNet, and SVHN samples.

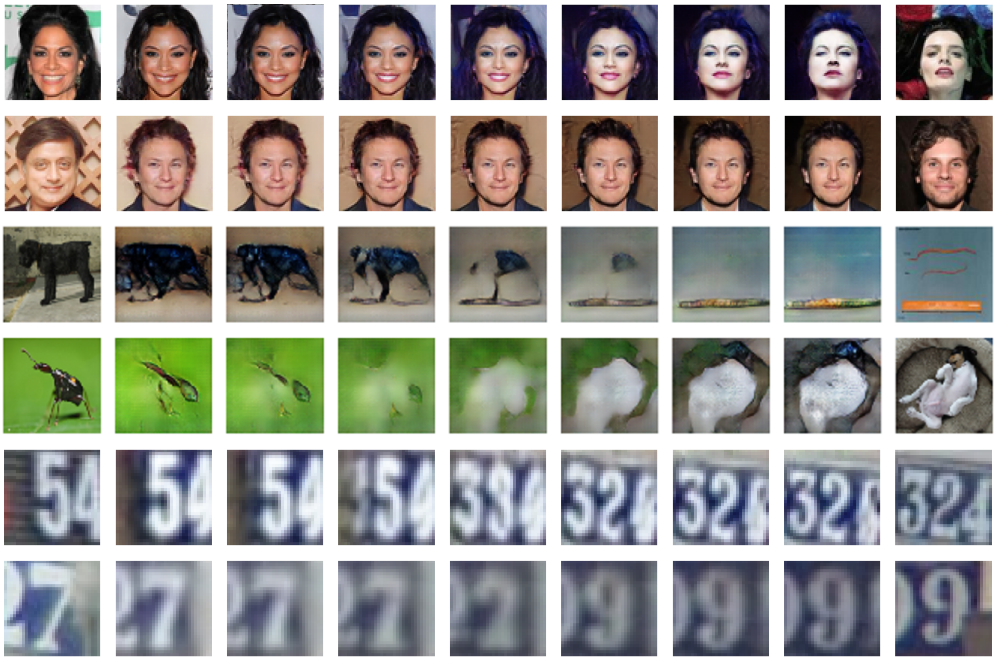

Figure 6: CelebA, ImageNet, and SVHN Reconstructions and Interpolations. The outermost images are originals, the adjacent images are reconstructions.

Method	Error rate
KNN (as reported in Zhao et al. (2015))	77.93%
TSVM (Vapnik, 1998)	66.55%
VAE (M1 + M2) (Kingma et al., 2014)	36.02%
SWWAE without dropout (Zhao et al., 2015)	27.83%
SWWAE with dropout (Zhao et al., 2015)	23.56%
DCGAN + L2-SVM (Radford et al., 2015)	$22.18\%(\pm 1.13\%)$
ALI (Dumoulin et al., 2016)	$19.14\%(\pm 0.50\%)$
SDGM (Maaløe et al., 2016)	$\mathbf{16.61\%(\pm 0.24\%)}$
IAN (ours)	$18.50\%(\pm 0.38\%)$

Table 1: Error rates on Semi-Supervised SVHN with 1000 training examples.

4 EXPERIMENTS

We evaluated IAN on CelebA (Liu et al., 2015), SVHN (Netzer et al., 2011) and Imagenet(Russakovsky et al., 2015).

4.1 SAMPLES

Samples from IAN, shown in Figure 5, display the high visual fidelity typical of adversarially trained networks.

4.2 RECONSTRUCTIONS AND INTERPOLATIONS

IAN demonstrates high quality reconstructions, shown in Figure 6, even displaying some semblance of similarity on ImageNet reconstructions. Interpolations are smooth and plausible, even between drastically different samples.

4.3 SEMI-SUPERVISED LEARNING WITH SVHN

We quantitatively evaluate the inference abilities of our architecture by applying it to the semi-supervised SVHN classification task. Our procedure follows that of (Radford et al., 2015) and (Dumoulin et al., 2016): We train an L2-SVM to classify SVHN data, using the output of the fully-connected layer of the encoder subnetwork as input features to the SVM. We report the average test error and standard deviation across 100 different SVMs, each trained on 1000 random examples from the training set. Our performance, as shown in Table 1, is competitive with other adversarial methods, achieving 18.5% mean classification accuracy.

5 CONCLUSION

We introduced the Neural Photo Editor, a novel interface for exploring the learned latent space of generative models and for making specific semantic changes to natural images. Our interface makes use of the Introspective Adversarial Network, a hybridization of the VAE and GAN that outputs high fidelity samples and reconstructions, and achieves competitive performance in a semi-supervised classification task. The IAN makes use of Multiscale Dilated Convolution Blocks and Orthogonal Weight Regularization, two techniques designed to improve model expressivity and training stability for adversarial networks.

ACKNOWLEDGMENTS

This research was made possible by grants and support from Renishaw plc and the Edinburgh Centre For Robotics. The work presented herein is also partially funded under the European H2020 Programme BEACONING project, Grant Agreement nr. 687676.

REFERENCES

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. Adversarially learned inference. arXiv Preprint arXiv: 1606.0070, 2016.

L.A. Gatys, A.S. Ecker, and M. Bethge. A neural algorithm of artistic style. arXiv Preprint arXiv: 1508.06576, 2015.

M. Germain, K. Gregor, I. Murray, and H Larochelle. Made: Masked autoencoder for distribution estimation. arXiv Preprint arXiv: 1502.03509, 2015.

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*, pp. 2672–2680, 2014.

K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. arXiv Preprint arXiv: 1603.05027, 2016.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML 2015*.

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

D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv Preprint arXiv: 1412.6980, 2014.

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

D.P. Kingma, T. Salimans, and M. Welling. Improving variational inference with inverse autoregressive flow. arXiv Preprint arXiv: 1606.04934, 2016.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images, 2009.

Alex Lamb, Vincent Dumoulin, and Aaron Courville. Discriminative regularization for generative models. *arXiv preprint arXiv:1602.03220*, 2016.

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

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3730–3738, 2015.

Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 4. Granada, Spain, 2011.

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

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. arXiv Preprint arXiv: 1606.03498, 2016.

A.M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *ICLR 2014*, 2014.

C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. arXiv Preprint arXiv: 1602.07261, 2016.

A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. arXiv Preprint arXiv: 1601.06759, 2016.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR 2016*, 2016.

Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann Lecun. Stacked what-where auto-encoders. *arXiv preprint arXiv:1506.02351*, 2015.