InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets

Explainable Machine Learning Report

by

Peter Hügel

Name:	Peter Hügel
Enrollment Number:	3140348
Supervisor:	PD. Dr. Ullrich Köthe

Contents

1 Generative Adversarial Network

This chapter will cover the fundamentals of a generative adversarial network (GAN) on which InfoGAN is built on. The method is explained in 1.1, in 1.2 a few results and applications are presented and discussed.

1.1 Approach

GAN was proposed in "Generative adversarial nets"[4] by Goodfellow, Pouget-Abadie, Mirza, Xu, Warde-Farley, Ozair, Courville, and Bengio. It is a method of estimating a generative model, a generator is trained to produce samples based on the training data X. While discriminative models are models of the conditional probability $P(Y|X = x)$, a generative model is a model of the conditional probabiility $P(X|Y = y)$.

In GAN the generator is trained with the help of an adversary, a discriminative model. Both models are multilayer perceptrons and essentially compete against each other. This competition results in the optimization of both parties. The generator G generates samples $\hat{x} = G(\mathbf{z})$ with a noise input vector $\mathbf{z}$. The noise vector is based on an arbitrary noise distribution $P_{noise}(\mathbf{z})$. The discriminator D tries to distinguish between the original data X and the data $\hat{X}$ that is generated by G. The outpout of $D(x)$ is the probability of x being from the true data distribution P_{data} rather than from the generators distribution $P_{generator}$. Therefore $D(x) \in [0, 1]$. Figure 1.1 displays the structure of GAN and the the two models.

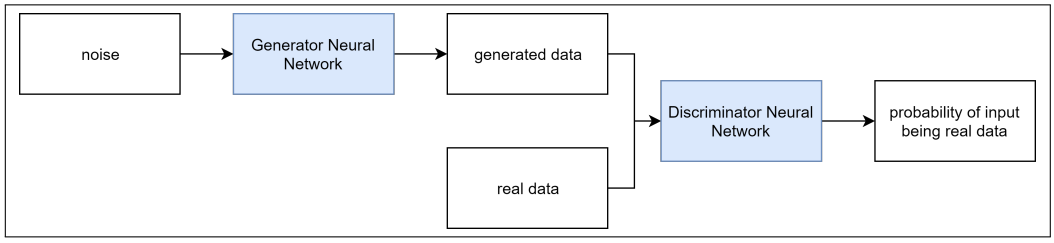


Figure 1.1: A display of the structure of GAN. Random noise is used to generate data with G. D then attempts to distinguish the generated data from the real data.

The real world analogy to this competition of the generator and the discriminator is a team of counterfeiters producing fake currency while investigators try to distinguish the fake money from real money. In this competition, the counterfeiters keep improving their method to fool the investigators. The investigators in turn also have to improve, so they can identify the fake money that keeps getting harder to distinguish from the real thing. This competition ends when the fake money and the real money are indistinguishable.

To implement this competition for the generator G and the discriminator D, they are optimized through the minimax game given by

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{data}}[\log D(x)] + \mathbb{E}_{\mathbf{z}\sim P_{noise}}[\log(1 - D(G(\mathbf{z})))]. \tag{1.1}$$

The generator G is trained by minimizing equation 1.1, the discriminator by maximizing it.

The first term in the equation becomes 0 when D perfectly distinguishes between real and fake data. In the case where $x \sim P_{data}$, the expected value of $D(x)$ is 1, therefore $\log D(x) = 0$. When D makes imperfect predictions, the expected value of $D(x)$ is smaller than 1, resulting in a negative expected value for $\log D(x)$.

The second term of the equation becomes 0 in the same scenario. In the case of $\mathbf{z} \sim P_{noise}$, meaning $G(\mathbf{z}) \sim P_{generator}$, the expected value of $D(G(\mathbf{z}))$ is 0 if the discriminator correctly identifies the data as being fake and 1 if the generated data is falsely classified as real. When perfectly identifying the generated data, the expected value for $(1 - D(G(\mathbf{z})))$ therefore is 1. The expected value of its logarithm $\log(1 - D(G(\mathbf{z})))$ is then 0. When the discriminator D is fooled by the generator G, the worst case is pure guessing, this results in an expected value of $\frac{1}{2}$ for $D(G(\mathbf{z}))$. So the second term becomes negative when G creates samples that D cannot distinguish from real data. In general, the optimal discriminator is $D(x) = \frac{P_{data}(x)}{P_{data}(x)+P_{generator}(x)}$.

In other words, the two models compete against each other in the second term. The generator attempts to minimize it while the discriminator tries to maximize it. The first term ensures that the discriminator does not simply classify all input as false data. Figure 1.2 displays the relevant distributions of GAN at different steps of the iterative process.

1.2 Results

In this section some interesting results that are generated with generative adversarial networks are discussed. In Figure 1.3 each of the ten rows displays the linear interpolation through the domain of P_{noise}, or Z, between two images that are generated with the trained generator $G(\mathbf{z})$. In essence, a traversal through the space of Z between two points is displayed in each row. Some interesting behaviour can be observed. It is important to note that all images that are created from the interpolated vectors in the Z-space can also be seen as proper bedrooms. Additionally they can be observed to smoothly transition within each row. Especially row six and row ten provide interesting insights. In row six a lamp can be seen to slowly transition into a window that grows in size. In row ten the same thing happens wit a TV until it can also be identified as a window. Similar scenarios can be observed in the other rows. Objects slowly transform when traversing in the Z-space.

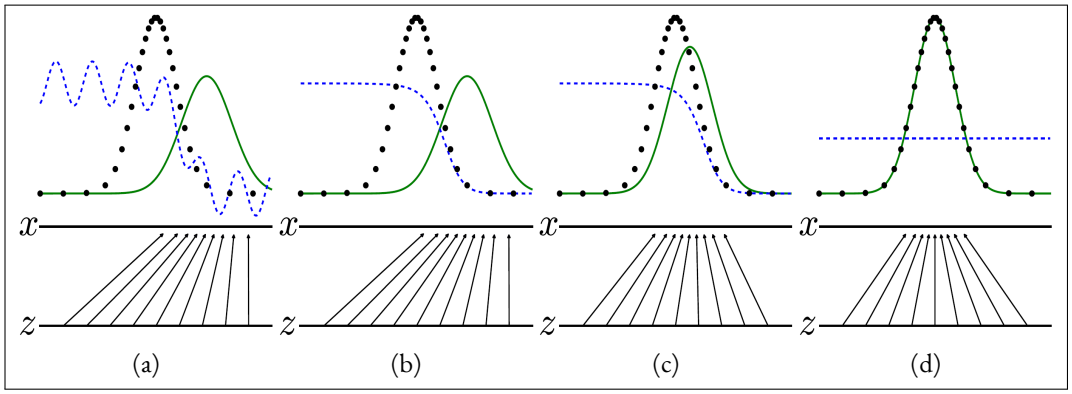


Figure 1.2: A representation of the distribution of the original data P_{data} (black dotted line) and the distribution of the generated data $P_{generator}$ (green line). Both distributions are displayed in the domain of X, the arrows indicate sampling from Z and mapping to X with $\hat{x} = G(\mathbf{z})$. The descriminators decision is also visualized (blue dashed line). (a) displays a step close to convergence in the iterative process. The distribution $P_{generator}$ overlaps with the real distribution P_{data}. The descriminator still distinguishes the the generated data fairly well. In (b) the discriminator is updated, converging to $\frac{P_{data}(x)}{P_{data}(x)+P_{generator}(x)}$. In (c) the generator G is updated and now generates a distribution that is a lot closer to the orignial data. In (d), after more steps, the generators distribution $P_{generator}$ has converged to the real data distribution P_{data}. The discriminators best option is now guessing as $\frac{P_{data}(x)}{P_{data}(x)+P_{generator}(x)} = \frac{1}{2}$.
IMAGE FROM "GENERATIVE ADVERSARIAL NETS"[4]

More interestingly, Figure 1.4 shows how vector arithmetic can be performed within the noise/feature space Z. in Figure 1.4(a) the mean noise vector of a neutral woman is subtracted from the mean noise vector of a smiling woman. The idea is to extract the vector responsible for adding the smile. Having obtained this vector, it is added to the noise vector of a neutral man. Interestingly enough, the resulting vector corresponds to images of a smiling man when fed into the generator $G(\mathbf{z})$. The same is done with the presence of glasses, in Figure 1.4(b) the mean noise vector of the existance of glasses in the image is extracted and used to create images of a woman with glasses. As the representations produced by GAN are in general highly entangled, this is not guaranteed to work. The vector that adds glasses to a man is not necessarily the same as the one that adds glasses to a woman.

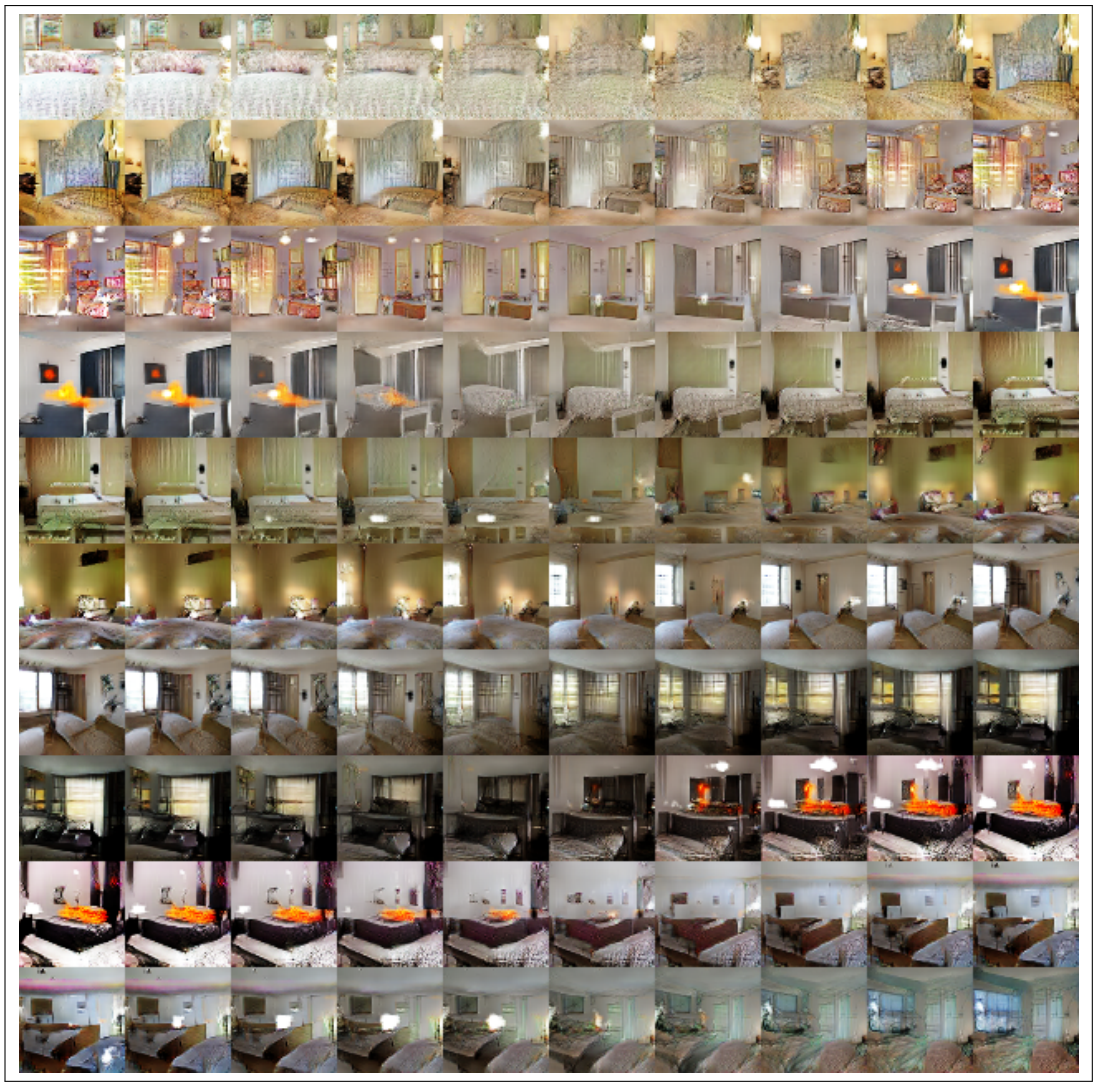


Figure 1.3: Each row is based on two generated images that are generated through $G(\mathbf{z})$ with the random noise vector $\mathbf{z} \sim P_{noise}$. These two images are the leftmost and rightmost image in each row. The images inbetween are generated by interpolating between the two vectors within the feature space Z and then generating images from the resulting noise vectors, essentially tracing a path through the feature space.
IMAGE FROM "UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS"[5]

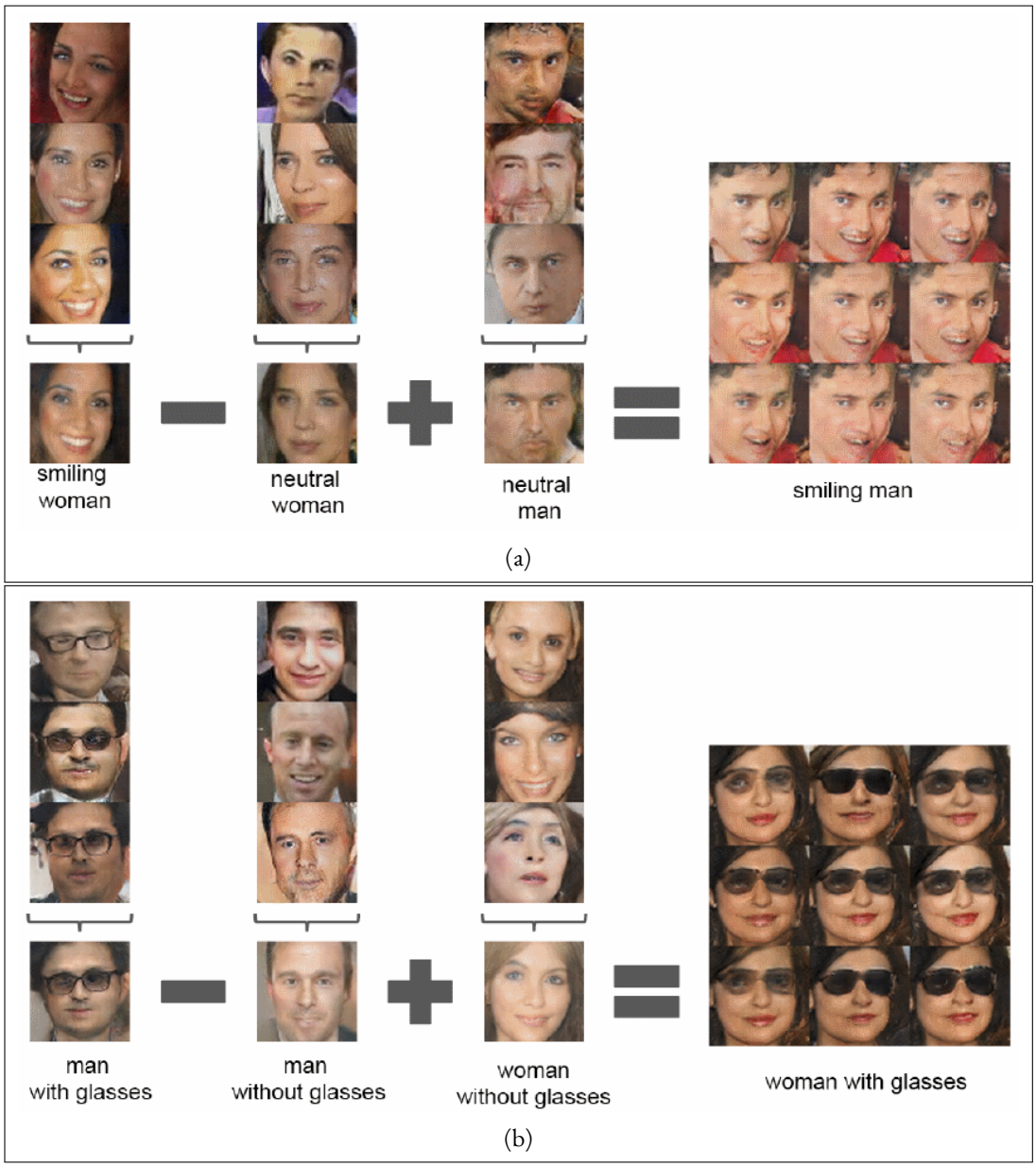


Figure 1.4: From a collection of generated images (collumns of three images), the mean vector within the feature space Z is calculated (indicated below those collumns). With these mean vectors, vector arithmetic is performed within the feature space. The resulting vector has some noise added to it to generate a collection of images (nine images on the right). (a) and (b) display the results for calculating the vectors within the feature space that correspond to a smiling man and a woman with glasses respectively.
IMAGE FROM "UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS"[5]

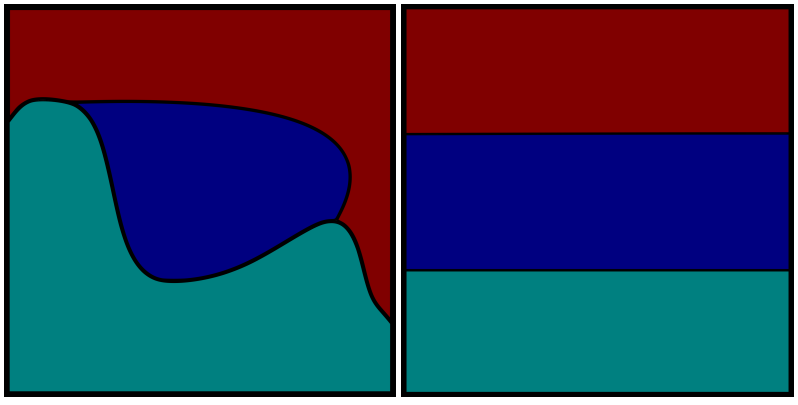
2 InfoGAN

This chapter starts with a brief explanation for the motivation of InfoGAN, which expands GAN, before elaborating on the approach and implementation. Afterwards a variety of results are displayed and discussed.

2.1 Motivation for InfoGAN

As briefly mentioned in Section 1.2, the representations generated by GAN are generally highly entangled. Instead of extracting vectors in the noise space Z that are responsible for specific features, having clear parameters for those features is desired. In other words, a disentangled representation of the feature space Z. In Figure 2.1 the idea of an entangled and a disentangled noise space is conveyed. In Figure 2.1(a) it can be hard to describe the exact areas of specific features as they are entangled. But in Figure 2.1(b) the same feature space is represented in a disentangled manner. The boundaries between relevant features are easily indentifyable and only dependant on one axis.

To bring this into perspective, applying this concept to the vector arithmetic performed in Figure 1.4, instead of extracting a vector for glasses, the area within the feature space that creates images of men and the area that creates images of women could be clearly seperated. The area for images of men could then be split into two areas, one for men with and one for men without glasses. In the entangled representation there could be multiple clusters with the same important feature that are spread out arbitrarily far.



(a) (b)

Figure 2.1: A visualization of a two-dimensional feature space. (a) is an entangled representation. (b) is a disentangled representation, the y axis can be observed to essentially be a feature dimension.

As the focus will lie on the MNIST dataset to explain the concepts of InfoGAN, Figure 2.2 displays the same concept of interpolating within the noise space that was used for the bedrooms in Figure 1.3. In Figure 2.2(a) and (b) a one slowly transitions into a five and a seven slowly transitions into a one. In the latter, the path through the feature space appears to pass an area of a nine.

The motivation for InfoGAN is the desire for a disentangled representation in the feature space that allows for the targeted generation of data with specific features. For example, generating only one type of digit with a generator trained on the MNIST dataset. While the digit type is the most obvious feature that can be distinguished, there are other features that may be of interest. For example the stroke thickness of the digits.

(a)

(b)

Figure 2.2: Interpolation of noise vectors $\mathbf{z}$ in the feature space from left to right. (a) displays a one being transformed to a five. (b) displays a seven transforming into a one.
IMAGE FROM "GENERATIVE ADVERSARIAL NETS"[4]

2.2 Approach

The idea with InfoGAN is to modify the input vector of the generator G. Instead of just using the noise vector $\mathbf{z}$ as in GAN, latent code is added to the noise vector. The generator now generates samples based on a combination of noise and user specified latent code $\mathbf{c}$. The generator therefore changes from $G(\mathbf{z})$ to $G(\mathbf{z}, \mathbf{c})$. In the case of MNIST, the latent code $\mathbf{c}$ is modeled to contain one uniform categorical code with ten possible categories. This is chosen because of the ten different digits present in MNIST. Additionally two uniform continuous codes are added.

The problem with normal GAN is that the latent code can be ignored by the network by finding the trivial solution of $P_{generator}(x|\mathbf{c}) = P_{generator}(x)$. To force the network to make use of the latent code, the minimax game given by Equation 1.1 is modified to ensure high mutual information between the latent code and the generated samples. Mutual information between two variables can be expressed through the change in entropy when gaining knowledge of one variable:

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X). \tag{2.1}$$

It can be observed that $I(X;Y)$ is zero when the entropy of X stays the same after gaining knowledge of Y, in other words there is no mutual information between X and Y.

For the case of the generator and the samples it generates, the mutual information between the latent code and the samples should be maximal. So $I(\mathbf{c}|G(\mathbf{z},\mathbf{c})$ should be maximal. In other words, knowing x should provide information about $\mathbf{c}$ and knowing $\mathbf{c}$ should provide information about x. So for any given $x \sim P_{generator}$, the entropy of $P_{generator}(\mathbf{c}|x)$ should be minimal. In the case of MNIST, an image of a certain digit should strongly correlate to one specific category of the categorical latent code.

As to retain this maximal mutual information, the minimax game as defined in GAN (see Equation 1.1) should be modified with an additional term:

$$\min_G \max_D V_I(D,G) = V(D,G) - \lambda I(\mathbf{c}; G(\mathbf{z},\mathbf{c})). \tag{2.2}$$

λ is a mere factor. With this term, the network is forced to make use of the latent code as maximal mutual information between the latent code and the produced samples is ensured.

In practice it is difficult to calculate $I(\mathbf{c}; G(\mathbf{z},\mathbf{c}))$ as it requires the posterior probability $P_{generator}(\mathbf{c}|x)$. Making use of variational information maximization [1], a lower bound for the mutual information is found with an auxiliary distribution $Q(\mathbf{c}|x)$:

$$\begin{aligned} I(\mathbf{c}; G(z,\mathbf{c})) &= H(\mathbf{c}) - H(\mathbf{c}|G(z,\mathbf{c})) \\ &= \mathbb{E}_{x\sim G(z,\mathbf{c})}[\mathbb{E}_{\mathbf{c}'\sim P(\mathbf{c}|x)}[\log P(\mathbf{c}'|x)]] + H(\mathbf{c}) \\ &= \mathbb{E}_{x\sim G(z,\mathbf{c})}[D_{KL}(P(\cdot|x)||Q(\cdot|x)) + \mathbb{E}_{\mathbf{c}'\sim P(\mathbf{c}|x)}[\log Q(\mathbf{c}'|x)]] + H(\mathbf{c}) \\ &\geq \mathbb{E}_{x\sim G(z,\mathbf{c})}[\mathbb{E}_{\mathbf{c}'\sim P(\mathbf{c}|x)}[\log Q(\mathbf{c}'|x)]] + H(\mathbf{c}) \end{aligned} \tag{2.3}$$

With this lower bound, the posterior probability does not need to be calculated explicitly, but a sampling from the posterior probability is still required in the inner expectation. To avoid this, Equation 2.3 can be further reformulated to define a variational lower bound L_I:

$$\begin{aligned} L_I(G,Q) &= \mathbb{E}_{\mathbf{c}\sim P(\mathbf{c}),x\sim G(z,\mathbf{c})}[\log Q(\mathbf{c}|x)] + H(\mathbf{c}) \\ &\geq I(\mathbf{c}; G(z,\mathbf{c})) \end{aligned} \tag{2.4}$$

This variational lower bound $L_I(G,Q)$ can easily be approximated by Monte Carlo simulation. $H(\mathbf{c})$ can be assumed to be constant with a fixed latent code distribution. As the auxiliary distribution Q approaches the real distribution $P(\mathbf{c}|x)$, the expected value of the ommited term in Equation 2.3 becomes zero:

$$\mathbb{E}_{x\sim G(z,\mathbf{c})}[D_{KL}(P(\cdot|x)||Q(\cdot|x))] \to 0 \tag{2.5}$$

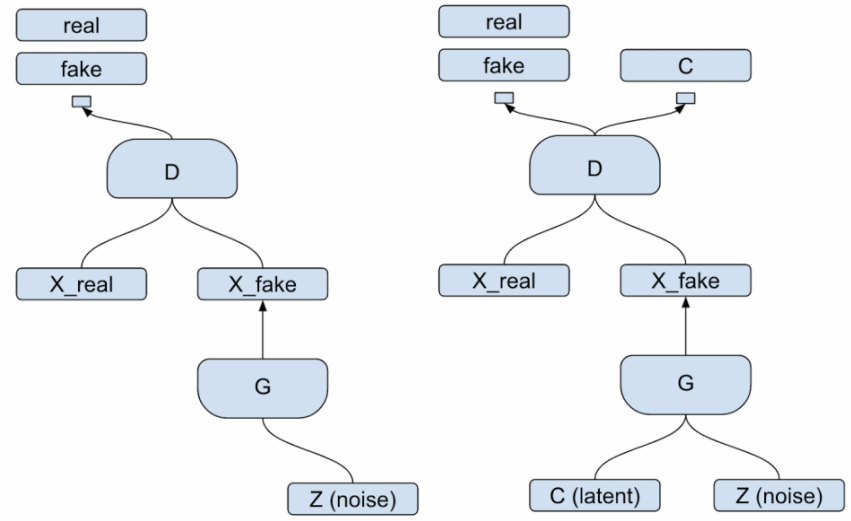
With this, the lower bound becomes tight. The final form of the modified minimax game is

$$\min_{G,Q} \max_D V_{\text{InfoGAN}}(D,G,Q) = V(D,G) - \lambda L_I(G,Q). \tag{2.6}$$

As the mutual information should be maximal, G and Q are trained by trying to maximize it in Equation 2.6. As the lower bound of the mutual information is subtracted, the whole term is to be minimized by G and Q. D has no direct impact on the mutual information. Considering

Equation 2.1, the mutual information will be between zero and $H(\mathbf{c})$. If the mutual information is equal to $H(\mathbf{c})$, it is maximal.

In practice, the auxiliary distribution Q does not require its own neural network and instead manifests in one additional fully connected layer that is added to the network of the discriminator D. The resulting structure of InfoGAN compared to normal GAN can be seen in Figure 2.3.



(a) (b)

Figure 2.3: (a) and (b) display the structure of GAN and InfoGAN respectively. The addition of latent code to the generator input can be observed while the distributors network is used to approximate the auxiliary distribution Q. In essence, $\mathbf{c}$ is estimated for each x (real or fake) through the network of D.
IMAGE FROM "UNDERSTANDING MUTUAL INFORMATION AND ITS USE IN INFOGAN"[3]

Figure 2.4 displays the value of the lower bound over many iterations. The green line visualizes the mutual information during GAN, when the network is not forced to make use of the latent code. The blue line shows the mutual information in InfoGAN, where the value function is modified to enforce high mutual information between the latent code and the generated samples. It can be observed that during InfoGAN the mutual information is quickly maximized and stays maximal. During GAN on the other hand, the latent code can be ignored, as mutual information is not enforced. For these results InfoGAN was used on the MNIST dataset with uniform categorical latent code $c \sim \text{Cat}(K = 10, p = 0.1)$.

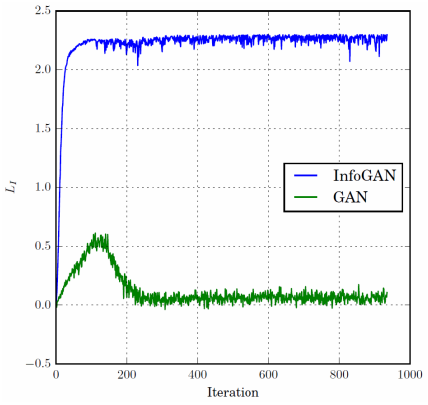


Figure 2.4: The lower bound L_I over many iterations for GAN (green) and InfoGAN (blue).
IMAGE FROM [2]

2.3 Results

This section will discuss the results generated by using InfoGAN on a variety of different datasets. Beginning with MNIST, the latent codes $\mathbf{c}$ were chosen to be the uniform categorical code $c_1 \sim \text{Cat}(K = 10, p = 0.1)$, and two uniform continuous latent codes $c_2, c_3 \sim \text{Unif}(-1, 1)$. InfoGAN correctly associates the digit type with the categorical latent code and finds two interesting features for the continuous code, a parameter for how much the digit is titled and a parameter for the line thickness. Having acquired this disentangled representation, the generator and the latent code can be used to generate specific images. For example a specific digit with a high line thickness that is specifically tilted. Figure 2.5 illustrates the results of InfoGAN on MNIST. While the continuous codes c_2 and c_3 ranged from -1 to 1 during training, to emphasize their effect, visualizations were made with it ranging from -2 to 2.

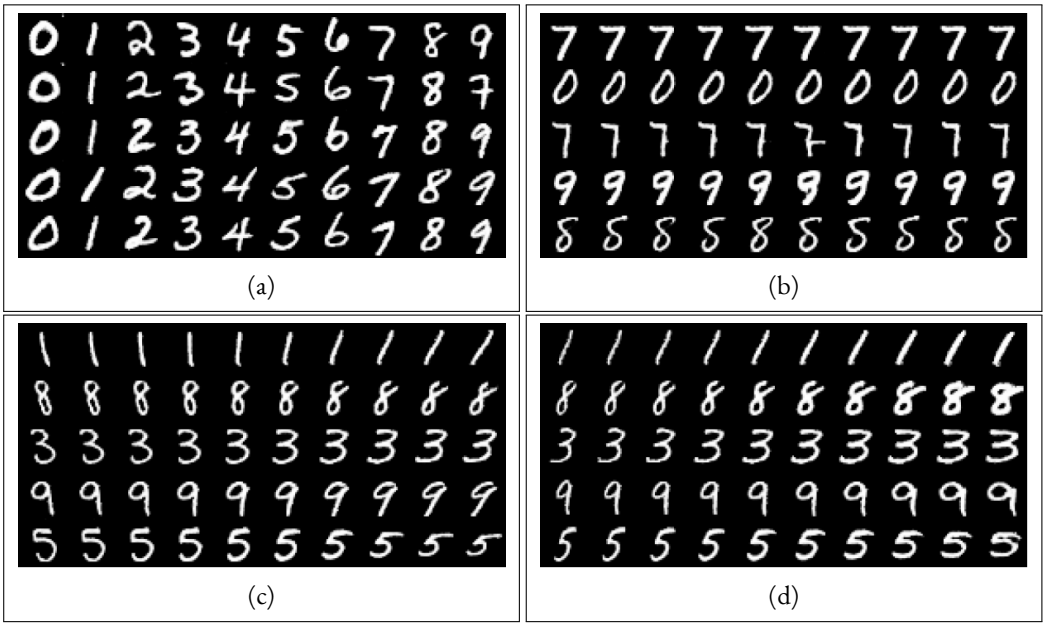


Figure 2.5: The noise z changes for each row while a specific latent code is gradually changed within the row from left to right. The noise and other latent codes are fixed within each row. (a) shows a varying c_1 (digit type). (b) shows a varying c_1 with regular GAN (no clear meaning). (c) shows a varying c_2 from -2 to 2 (rotation). (d) shows a varying c_3 from -2 to 2 (line thickness). IMAGE FROM "INFOGAN: INTERPRETABLE REPRESENTATION LEARNING BY INFORMATION MAXIMIZING GENERATIVE ADVERSARIAL NETS"[2]

InfoGAN was also tested on 3D datasets of faces and chairs (see Figure 2.6 and 2.7). In both cases interesting features have been recovered. For the dataset of faces, the pose, elevation, lighting and width are extracted. On the chairs dataset, rotation and width are recovered. Figure 2.8 shows results on a dataset of housenumbers. In Figure 2.9, for an unlabeled dataset of celebrity faces, ten latent codes were chosen that each have ten dimensions.

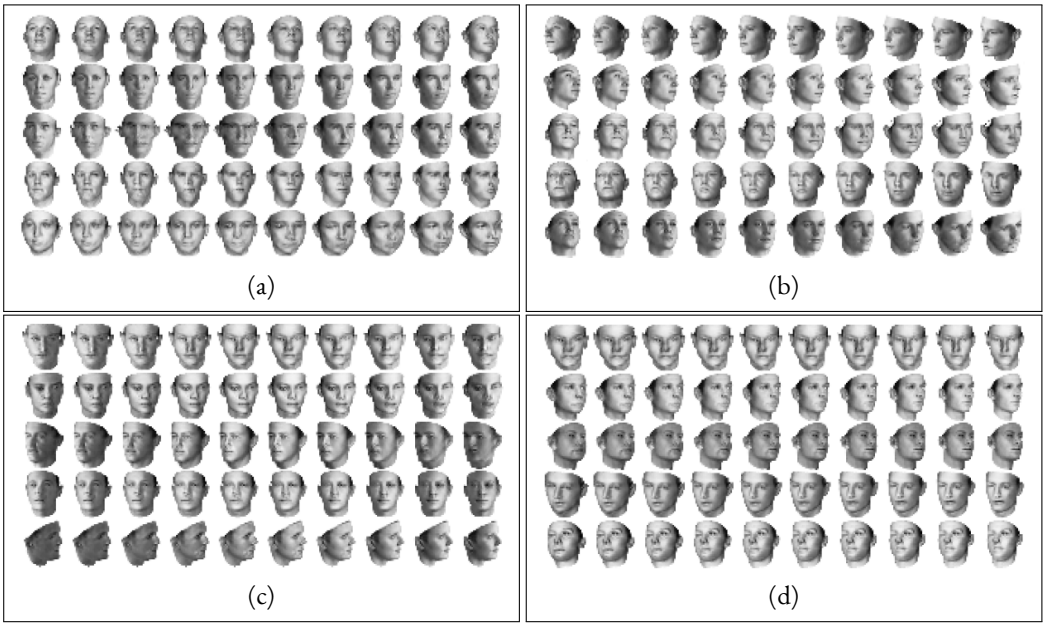
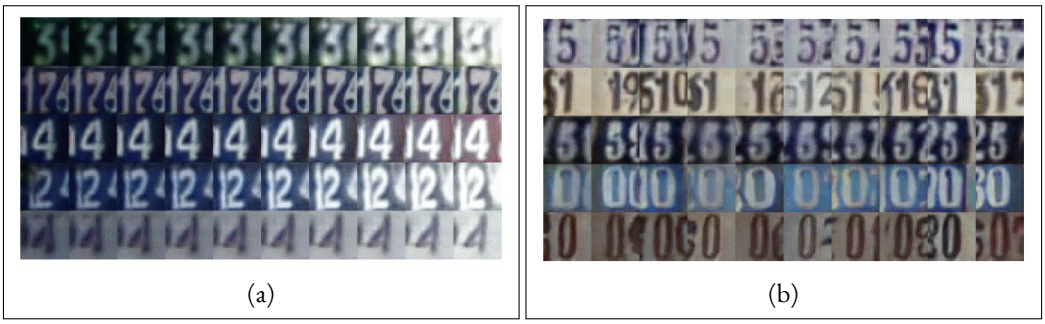


Figure 2.6: The noise z changes for each row while a specific latent code is gradually changed within the row from left to right. The noise and other latent codes are fixed within each row. For this dataset, the latent codes were modeled with five continuous codes.
(a): azimuth (pose) (b): elevation (c): lighting (d): wide or narrow
IMAGE FROM "INFOGAN: INTERPRETABLE REPRESENTATION LEARNING BY INFORMATION MAXIMIZING GENERATIVE ADVERSARIAL NETS"[2]

Figure 2.7: The noise z changes for each row while a specific latent code is gradually changed within the row from left to right. The noise and other latent codes are fixed within each row. For this dataset, the latent codes were modeled with four categorical 20-dimensional codes and one continuous code.
(a): rotation (b): width
IMAGE FROM "INFOGAN: INTERPRETABLE REPRESENTATION LEARNING BY INFORMATION MAXIMIZING GENERATIVE ADVERSARIAL NETS"[2]



(a) (b)

Figure 2.8: The noise z changes for each row while a specific latent code is gradually changed within the row from left to right. The noise and other latent codes are fixed within each row. For this dataset, the latent codes were modeled with four categorical 10-dimensional codes and two continuous codes.
(a): lighting (b): context (categorical latent code)
IMAGE FROM "INFOGAN: INTERPRETABLE REPRESENTATION LEARNING BY INFORMATION MAXIMIZING GENERATIVE ADVERSARIAL NETS"[2]

(a) (b) (c) (d)

Figure 2.9: The noise z changes for each row while a specific latent code is gradually changed within the row from left to right. The noise and other latent codes are fixed within each row. For this dataset ten categorical latent codes were chosen, each having ten dimensions.
(a): azimuth (pose) (b): presence of glasses (c): hair style (d): emotion
IMAGE FROM "INFOGAN: INTERPRETABLE REPRESENTATION LEARNING BY INFORMATION MAXIMIZING GENERATIVE ADVERSARIAL NETS"[2]

3 Conclusion

Chen, Duan, Houthooft, Schulman, Sutskever, and Abbeel successfully implemented a representation learning algorithm that is based on GAN. Previous approaches have required supervision while InfoGAN can operate on unlabeled data. As unlabeled data is plentiful, this new method proves highly useful. While parameters are provided in the form of the latent codes, they are not required to be connected to specific features manually. Instead, with InfoGAN the most relevant features are extracted. With the optimizations of the initial approach, InfoGAN comes with little to no cost compared to normal GAN. Interpretable representations have been achieved as shown in Section 2.3. Compared to GAN, InfoGAN seems to provide obvious benefits. While the vector arithmetic in the feature space enabled the calculation of some features in relation to others, InfoGAN delivers a reliable disentangled representation in which the feature space is spanned by relevant features.

Bibliography

1. D. B. F. Agakov. "The IM algorithm: a variational approach to information maximization". *Advances in Neural Information Processing Systems* 16, 2004, p. 201.
2. 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*. 2016, pp. 2172–2180.
3. K. Evtimova and A. Drozdov. "Understanding Mutual Information and its Use in InfoGAN".
4. 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, pp. 2672–2680.
5. A. Radford, L. Metz, and S. Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". *arXiv preprint arXiv:1511.06434*, 2015.