UNSUPERVISED LEARNING USING GENERATIVE ADVERSARIAL TRAINING AND CLUSTERING

Vittal Premachandran and Alan L. Yuille
Department of Computer Science
Johns Hopkins University
{vittalp, ayuille1}@jhu.edu

ABSTRACT

In this paper, we propose an unsupervised learning approach that makes use of two components; a deep hierarchical feature extractor, and a more traditional clustering algorithm. We train the feature extractor in a purely unsupervised manner using generative adversarial training and, in the process, study the strengths of learning using a generative model as an adversary. We also show that adversarial training as done in Generative Adversarial Networks (GANs) is not sufficient to automatically group data into categorical clusters. Instead, we use a more traditional grouping algorithm, k-means clustering, to cluster the features learned using adversarial training. We experiment on three well-known datasets, CIFAR-10, CIFAR-100 and STL-10. The experiments show that the proposed approach performs similarly to supervised learning approaches, and, might even be better in situations with small amounts of labeled training data and large amounts of unlabeled data.

1 INTRODUCTION

Much of the recent work in machine learning and computer vision has focused on llearning techniques for high-level tasks such as image classification (Krizhevsky et al. (2012); Simonyan & Zisserman (2014); He et al. (2015)). Many of the state-of-the-art models employ Convolutional Neural Networks (CNNs) to extract high-level feature representations by processing the input data using multiple layers of convolutions, usually followed by some non-linear transform. CNNs have successfully demonstrated to yield high-quality feature representations that produce state-of-the-art results on a variety of tasks, not only on image classification (as mentioned above), but also on semantic segmentation (Long et al. (2015); Chen et al. (2016a)), boundary detection (Xie & Tu (2015); Premachandran et al. (2015)), and object detection (Girshick et al. (2014)), among others. These models are trained to produce high-quality features using backpropagation, usually by pretraining on a large dataset (such as ImageNet) and then fine tuning on the relevant dataset. Unfortunately, supervised learning suffers from certain challenges, especially, in terms of scalability since it requires large amounts of labeled data. Labeling millions of images requires extensive effort and is time consuming. Moreover, supervised training with a predefined set of classes, limits the generalizability of the learned feature representations to novel classes.

To overcome the difficulties of labeling large amounts of training data, effort has gone into the development of semi-supervised and unsupervised learning techniques. The goal of unsupservised learning techniques is to learn representations that are interpretable, easily transferable to novel tasks and novel object categories, and to disentangle the informative representation of the data from nuisance variables (e.g. lighting, viewpoint, etc.) purely from unlabeled data. A common and widely used method for unsupervised learning is to do clustering using k-Means. k-Means clustering is a simple method that groups input features into different clusters. Traditionally, this approach mainly used low-level features such as raw pixel intensities, HOG features, GIST features, SIFT features, etc. Although the performance of k-means on such features is usually poor, Wang et al. (2015) used deep network features and employed k-means clustering to show strong results on grouping object parts. But, the deep network that was used to extract the features was pre-trained on ImageNet using class-label supervision (so, object knowledge was known). It would be a natural extension to see if one can learn robust features using hierarchical feature learning in a purely unsupervised manner.

However, since the objectives of unsupervised learning are not as concrete as the objectives of supervised learning, optimizing deep hierarchical models using backpropagation becomes difficult.

Attempts have been made to come up with “pretext” objective functions, which are usually driven by “common sense” requirements, to do unsupervised learning. Some examples of these objectives include minimizing the reconstruction error (Vincent et al. (2008)), training models to identify surrogate classes (Dosovitskiy et al. (2014)), predicting spatial position of image patches (Doersch et al. (2015); Noroozi & Favaro (2016)), and minimizing the distance in the representation space for objects tracked over a time period in a video sequence (Wang & Gupta (2015))

Recently, much interest has gone into adversarial training. Generative Adversarial Networks (GANs) (Goodfellow et al. (2014)) are of particular interest in this work. Progress in GANs have enabled significant improvement in the quality of images being generated in the past couple of years (Denton et al. (2015); Radford et al. (2015)). While much of the recent effort has gone in the development of better architectures and training procedures for modeling and training the generative network, in this work, we systematically study the power of the representations learned by the generator’s adversary, i.e., the discriminative model.

In this paper, we learn a deep network using generative adversarial training. We use the features extracted from the discriminative component and fuse it with traditional unsupservised learning algorithms like k-Means to improve their performance. We perform various experiments over many different datasets (CIFAR-10, CIFAR-100 and STL-10) and show that the representations that can be learned purely by unsupervised learning from an adversarial signal helps to learn meaningful representations of input data. Our experiments show that under situations with minimal amounts of supervised training examples (and large amounts of unsupervised data), the representations learned with adversarial training perform competitively in comparison to supervised training on a similar architecture. We now provide a brief summary of adversarial training employed by GAN and InfoGAN.

2 Background on Adversarial Training

Generative Adversarial Networks (Goodfellow et al. (2014)) are composed of two components; the generator, $G(.)$, and the discriminator, $D(.)$. The generator maps a latent encoding to the data space, while the discriminator distinguishes between samples generated by the generator and real data. The generator is trained to fool the discriminator, while the discriminator is trained to not get fooled by the generator.

More formally, given training data samples, $\mathbf{x} \sim P_{data}(\mathbf{x})$, where $P_{data}(\mathbf{x})$ is the true data distribution, the training of GANs proceeds by iterating between two-steps. In the first step, we fix the parameters of the generative model, sample a latent code, $\mathbf{z} \sim P_{noise}(\mathbf{z})$, and generate data samples, $G(\mathbf{z})$, which is then used to train the discriminator, $D(.)$, by updating its parameters to distinguish between $G(\mathbf{z})$ and $\mathbf{x}$. The parameters of the discriminator can be updated by maximizing the expected log-likelihood,

$$\mathbb{E}_{\mathbf{x}\sim P_{data}(\mathbf{x})}[log(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim P_{noise}(\mathbf{z})}[log(1 - D(G(\mathbf{z})))]. \quad (1)$$

In the second step, we fix the parameters of the discriminator and update the parameters of the generator to generate samples that get classified as real by the discriminator. The parameters of $G(.)$ can be updated by minimizing,

$$\mathbb{E}_{\mathbf{z}\sim P_{noise}(\mathbf{z})}[log(1 - D(G(\mathbf{z})))]. \quad (2)$$

The objective of this minimax game can be written as

$$\min_G \max_D V(G, D) = \mathbb{E}_{\mathbf{x}\sim P_{data}(\mathbf{x})}[log(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim P_{noise}(\mathbf{z})}[log(1 - D(G(\mathbf{z})))]. \quad (3)$$

2.1 InfoGAN

The formulation described above uses a noise vector, $\mathbf{z}$, which is used by the generator, G(.), to synthesize data. This noise vector does not impose any constraints on what the generated data should look like. Chen et al. (2016b) introduce a neat and simple idea to extend GANs into a feature identifying system called InfoGAN. InfoGAN uses a structured latent code, $\mathbf{c}$, which is input to the generator, G(.), in addition to the noise vector, $\mathbf{z}$. The code can either be a discrete code or a continuous code. In order to encourage the code to capture the inherent semantic structures in the training data, a new term is introduced to the objective function, which acts as a regularizer that forces high mutual information between the latent code, $\mathbf{c}$ and the generated sample, $G(\mathbf{z}, \mathbf{c})$. Since it is hard to maximize the mutual information, $I(\mathbf{c}; G(\mathbf{z}, \mathbf{c}))$, directly (because one would need to know the true distribution $P(\mathbf{c}|\mathbf{x})$), Chen et al. (2016b) provide a variational lower bound, which can be obtained when using a parametric auxiliary, $Q(\mathbf{c}|\mathbf{x})$, to approximate $P(\mathbf{c}|\mathbf{x})$. The variational lower bound that is obtained is,

$$L_I(G, Q) = \mathbb{E}_{\mathbf{c}\sim P(\mathbf{c}), \mathbf{z}\sim P_{noise}(\mathbf{z})}[\log Q(\mathbf{c}|G(\mathbf{c}, \mathbf{z}))] + H(\mathbf{c}). \tag{4}$$

The InfoGAN objective is a regularized version of the original GAN objective (Eq. 3), where the regularizer is the variational lower bound of mutual information,

$$\min_{G,Q} \max_{D} V_{InfoGAN}(G, D, Q) = V(G, D) - \lambda L_I(G, Q). \tag{5}$$

Chen et al. (2016b) share the parameters between Q(.) and D(.), which helps reduce the computational cost. We do the same in all of our experiments.

As can be seen from the first term of Eq. 4, the lower bound of the mutual information regularizer conveniently turns out to be a recognition model. If the optimization procedure converges successfully, one can hope to have learned a latent code that ends up representing the most salient and structured semantic features present in the data. The noise parameters, $\mathbf{z}$, end up providing the stochasticity to the input that result in the production of samples with diversity.

3 Unsupervised Learning With Adversarial Training and k-Means++ Clustering

As mentioned in Section 1, we are interested in learning representations of images in a purely unsupervised manner. Both GAN, and InfoGAN provide a way to train the discriminative network using the generated images as an adversary. InfoGAN, is particularly interesting since it has the ability to directly predict the different categories that might be present in the training database. While the qualitative results presented in Chen et al. (2016b) shows that the categories can be automatically identified on the MNIST dataset, unfortunately, the same result does not seem to extend to more complicated and realistic datasets (CIFAR-10, CIFAR-100 and STL-10). We modified the InfoGAN code released by the authors to enable support of the more realistic RGB data. We then trained the model on the above mentioned datasets to experiment if it could automatically identify the categorical clusters present in the respective datasets. We found that while InfoGAN that we trained on the above-mentioned datasets was successful in generating images that looked different for different categorical codes, it was unable to identify the class-level grouping that is present in these datasets.

Instead, we adopt a hybrid strategy for unsupervised learning. We first use the generative network as an adversary to train the discriminative network until convergence. Upon convergence, we extract features from the penultimate layer of the D(.) network and run a more traditional clustering algorithm, i.e., k-means++. Surprisingly, this simple strategy turns out to be much more effective at grouping data from similar categories than the approach of directly predicting the categorical groups. Note that one can plug in more sophisticated unsupervised learning algorithms instead of k-means++. We use k-means++ to show that even a simple approach can produce reasonable results.

Another motivation for using the features from the penultimate layers is that it facilitates feature transferability to novel classes and tasks. It is common in the supervised learning approaches to first train a deep network on ImageNet images using class-level supervision, then to perform net surgery to chop off the top level weights, and using this truncated network as a feature extractor for further fine tuning on different datasets and tasks. Doing so does not prevent the model from being trained only on the ultimate task that it might be used for. One can train the network on a “pretext” task and transfer the learned weights to other novel tasks. This is especially crucial for unsupervised learning since the pretext task that is used to train the models is almost always much different from the specific task that the model will ultimately be used for.

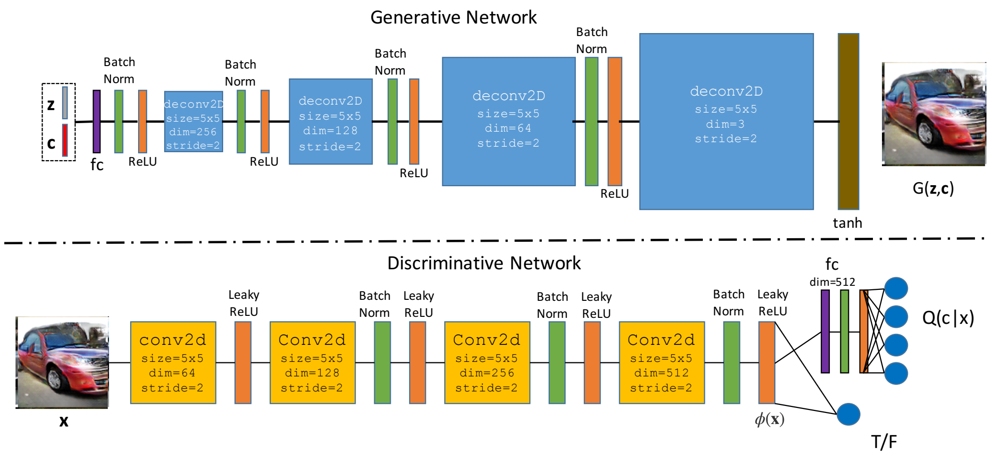

Generative Network
z
c
fc
Batch Norm
ReLU
deconv2D size=5x5 dim=256 stride=2
Batch Norm
ReLU
deconv2D size=5x5 dim=128 stride=2
Batch Norm
ReLU
deconv2D size=5x5 dim=64 stride=2
Batch Norm
ReLU
deconv2D size=5x5 dim=3 stride=2
tanh
G(z,c)
Discriminative Network
x
conv2d size=5x5 dim=64 stride=2
Leaky ReLU
Conv2d size=5x5 dim=128 stride=2
Batch Norm
Leaky ReLU
Conv2d size=5x5 dim=256 stride=2
Batch Norm
Leaky ReLU
Conv2d size=5x5 dim=512 stride=2
Batch Norm
Leaky ReLU
$\phi(\mathbf{x})$
fc dim=512
Q(c|x)
T/F

Figure 1: Figure shows the InfoGAN architecture that was used in all our experiments. Notice that the input to G(.) is a combination of **z** and **c**. Also notice that most of the parameters are shared between the Q(.) network and the D(.) network, thus improving the computational efficiency.

3.1 Network Architecture

We use the DCGAN architecture from Radford et al. (2015) since it is widely used for generating images. Figure 1 shows a visualization of the architecture.

Generator: Note that the generator has been slightly modified to accept the structured latent code, **c**, in addition to the random noise, **z**. The first layer is a `fully-connected (fc)` layer, which is then reshaped into a 2-D grid of spatial resolution $s/16 \times s/16$, where s is the size of the output image to be produced. Subsequent to this reshaping, the architecture has four layers of `transposed_convolution` (sometimes referred to as deconvolution) with a stride of 2, each of which upsamples the input features to twice the spatial resolution. These layers are sandwiched by `batch_norm` and `ReLU` layers. Finally, we use a `tanh` non-linearity to map the features into $[-1, 1]$.

Discriminator: The discriminator is a standard CNN with a series of convolutional layers followed by non-linearities. The architecture uses four convolutional layers sandwiched by `batch_norm` and `leakyReLU` layers. We don't use `max_pooling` to reduce the spatial resolution of the input. Instead, we convolve the feature maps with a stride of two, which results in the output of each convolution layer to be half the spatial resolution of the input feature map. This base architecture is shared between D(.) and Q(.). On top of this shared network, we use an `fc` layer to extract the features, which are then used to predict the categorical distribution. Notice that most of the computational cost is shared between the D(.) and the Q(.) networks thereby making the entire training process to be computationally efficient.

3.2 Unsupervised Learning with k-Means++

As mentioned previously, while InfoGAN has the ability to group data into multiple groups automatically, there is no constraint to enforce that the groups need to correspond to the various object-level categories that are present in the dataset. While this turned out to be true for the MNIST dataset (Chen et al. (2016b)), we believe that it was possible because the variations in the strokes that produce different digits correspond to the source of biggest variation in the dataset, which conveniently corresponds to the various digit categories, thereby enabling InfoGAN to act as a category recognition model. In more realistic datasets, the sources of biggest variation need not (and, usually, do not) correspond to variations in the object-level categories. Our experiments show this to be true. When we trained InfoGAN to automatically group the CIFAR-10 images into 10 categories, we found that while InfoGAN was able to group the images into different groups, the groups did not correspond to object category-level groupings. Figure 2 shows some example samples generated by the model.

Each row corresponds to a different category and each column in the row corresponds to a different sample from that category (obtained by keeping $\mathbf{c}$ fixed and by varying $\mathbf{z}$). We can see that while each row look different from each other, it does not correspond to the CIFAR-10 categories.

Therefore, we employ a hybrid approach to unsupervised clustering. We first train the discriminative network using either the vanilla GAN objective or the InfoGAN objective, until convergence. Upon convergence, we extract features for each image in the training set, from the top of the shared network, labeled as $\phi(\mathbf{x})$ in Figure 1, and do `average_pooling` across the spatial resolution, for each feature channel. We then cluster these features using k-means++ into a discrete set of k-categories. We set k to be the number of object classes that are present in the respective dataset. The cluster centers learned by k-means++ clustering act as the templates for the k categories that are present in the dataset.

During testing, we extract the feature representation of the test images by passing them through the discriminative network trained using the generator as an adversary, do `average_pooling` on $\phi(\mathbf{x})$, and compute the distance of the test feature vector to each of the centers learnt by k-means++ clustering during the training phase. The test image is assigned an index corresponding to the index of the closest center. Our experiments show that clustering on $\phi(\mathbf{x})$ produces better results than directly using the recognition model of InfoGAN. Note that while we use the simple k-means++ algorithm for clustering, it could be replaced by more sophisticated unsupervised learning algorithms. We do not explore further down this route since the scope of this work is to study the strength of the features learned by adversarial training.

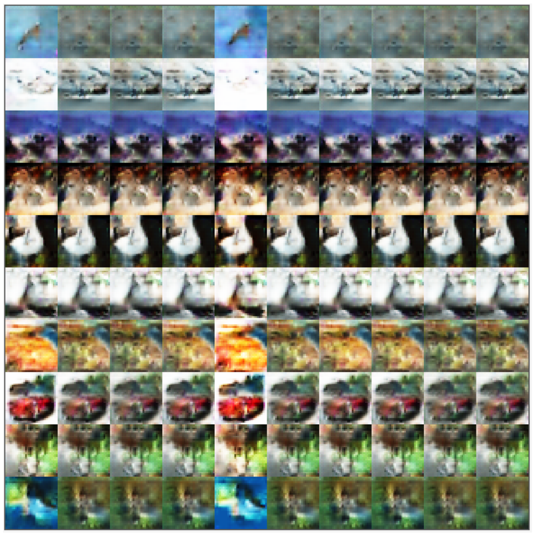


Figure 2: Figure shows samples generated from InfoGAN trained on the CIFAR-10 dataset when the system was encouraged to identify 10 categories. Each row corresponds to a different cluster identified by InfoGAN. Each column corresponds to a different sample from that clusters. We can see that while InfoGAN can identify clusters that are different from each other, they do not correspond to the CIFAR-10 categories. See Sec. 4.1 for quantitative results.

An advantage of the hybrid approach is that it now allows us to use a variety of different “pretext” objectives. In other words one can decouple the training objective from the testing requirements. In fact, we experimented by encouraging InfoGAN to identify more groups in the training data than number of object-categories in the dataset. For example, we trained InfoGAN on CIFAR-10 dataset by encouraging the system to identify [10, 20, 30, 35, 40, 50 and 75] groups. Of course, these groups do not correspond to category-level groupings. However, to our surprise, we found that when the features obtained from InfoGANs trained on large number of categories were used for clustering, they performed better at object categorization than the features obtained from an InfoGAN trained on the same number of object categories as present in the dataset. Section 4 provides quantitative results on these experiments.

4 Experiments

We perform experiments on multiple datasets; CIFAR-10, CIFAR-100 and STL-10[1]. We use ground truth labels only for evaluation purposes and for training the supervised learning baseline. The training procedure is entirely unsupervised. We report results using two standard metrics that are used for evaluating unsupervised learning algorithms; Adjusted RAND Index (ARI) and the Normalized Mutual Information (NMI) score. We provide three baselines; (i) we report results using simple features such as pixel intensities, HOG and GIST, which we call low-level visual features, (ii) we report results on the features obtained using standard GAN training, (iii) as an upper bound, we report results using supervised learning where we train the weights in a discriminator network with the same architecture using category-level labels that are provided by the datasets.

It is important to remember that we are interested in comparing the quality of the learned features that can be used for transfer to novel images and not just the classification score on an pre-defined set of categories. The classification accuracy captures only how well a test image was correctly classified. If incorrectly classified, it does not quantify how bad the mistake was. ARI, on the other hand, is a better metric for evaluating the properties of the features because it measures not only how accurately pairs of objects were correctly grouped together, but also takes into account how many pairs of data points were incorrectly grouped. Therefore, when comparing with the model that was trained using supervised learning, we ignore the top-level classification layer of that model, and quantify the quality of the representations, i.e., the features extracted from the penultimate layer, using ARI after clustering on them.

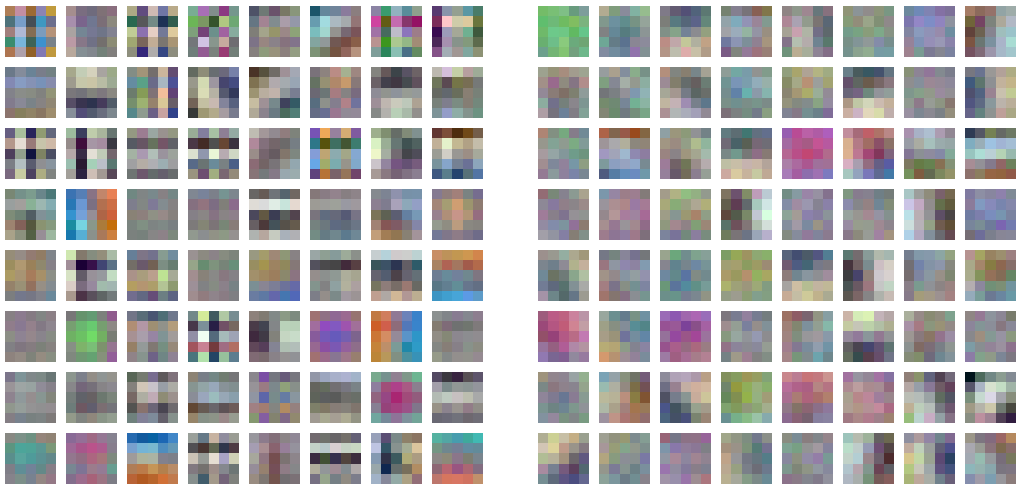
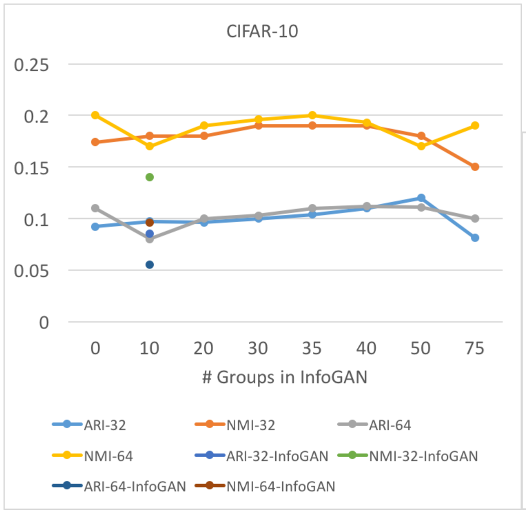


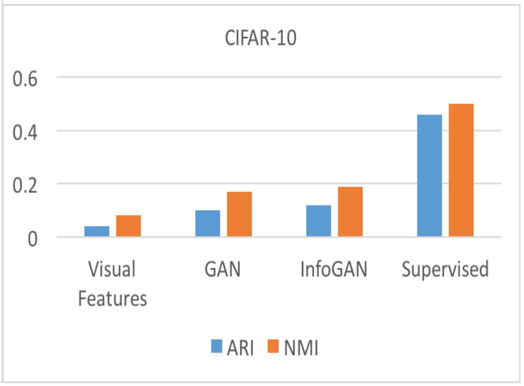
Figure 3: This figure shows all the 64 filters from the first layer of the discriminative network trained on CIFAR-10. The visualization on the left corresponds to the filters learned using adversarial training. The visualization on the right corresponds to the filters learned for the same architecture using supervised learning. It is interesting to see that there the filters on the left have more high frequency components and the filters on the right are more smooth.

Before we go into the quantitative results, we visualize the filters of the first layer of the discriminative network and compare them across two different training procedures. Figure 3 shows the visualization. On the left are the filters from the network that was trained using adversarial training. On the right are the filters from a network with the same architecture but trained using class-level supervision. Both these networks were trained using the CIFAR-10 dataset. We can see that while some of the filters look similar to each other, many of them are quite different. It is clear that the filters on the right are more smooth than the filters on the left. Recollect that filters on the left are trained to fit both the real images and the generated images. When the generated images are not as high-quality as the real images, the filters that D(.) learns might not be as regularized as the ones

[1]We have released the code that was used in all our experiments at https://github.com/VittalP/UnsupGAN



(a)



(b)

Figure 4: CIFAR-10: (a) Plots the performance of the grouping algorithm when using the features learned from InfoGAN training when trained over multiple categories. Zero groups corresponds to vanilla GAN. -32 and -64 correspond to the output sizes of the generated images. -InfoGAN corresponds to the results obtained with direct prediction using the recognition model in InfoGAN. (b) Note that InfoGAN features perform better than vanilla GAN features. However, supervised learning outperforms unsupervised learning on this database.

learnt using only real data. We hypothesize that improving the quality of the generated images can help regularize the first layer filters in D(.). We leave this route of exploration for future work.

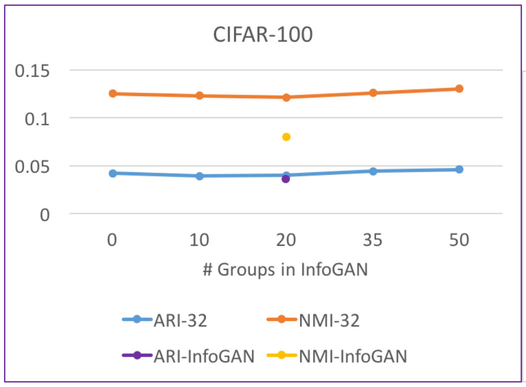
4.1 CIFAR-10

The CIFAR-10 consists of 50k training images and 10k testing images, of size 32×32, divided among 10 categories. We trained the model for two different image sizes; 32×32 and 64×64. We trained InfoGAN with different numbers of categories $\{10, 20, 30, 35, 40, 50, 75\}$. Figure 4a shows a plot of the performance measures versus the number of groups InfoGAN was trained to identify. We can see from the figure that as we increase the number of categories, the performance of the model goes up into a certain point and drop after that. This indicates that there exists databases for which grouping into more categories than present in the ground truth might help. We also plot the performance of the InfoGAN model when used directly as a prediction model. We can see from the plots that k-means++ clustering produces better results (ARI-32=0.097; NMI-32=0.18) than direct prediction (ARI-32-InfoGAN: 0.085; NMI-32-InfoGAN: 0.14). We label the direct prediction results with a (-InfoGAN).

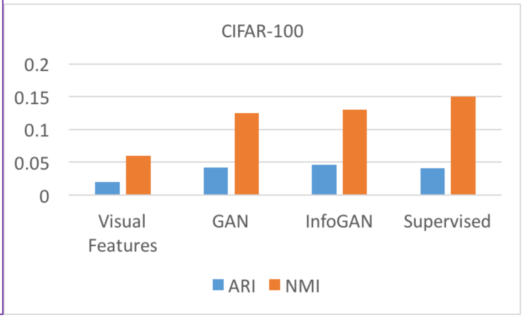
Figure 4b compares the performance when using different features. We can see that InfoGAN features trained with 50 clusters beats the features learned using vanilla GAN by a small margin. However, supervised training does much better (as one might have expected).

4.2 CIFAR-100

In these sets of experiments, we use the images from the CIFAR-100 database for training. This database also contains 50k training examples and 10k test images, divided among 100 fine scale categories and 20 coarse level categories. We test the performance on the coarse categories. As before, we experiment the InfoGAN training with multiple categories $\{10, 20, 35, 50\}$. While the trend is not as noticeable as in the case of CIFAR-10, the best performance is obtained when we use 50 categories. Also, as before, the k-means++ clustering of the features produces better performance (ARI=0.04) than the recognition model of InfoGAN (ARI=0.036).



(a)



(b)

Figure 5: CIFAR-100: (a) # of groups used to train InfoGAN has less of an effect on CIFAR-100 than it had on CIFAR-10. However, the performance of k-means++ clustering is still better than direct prediction using the recognition model of InfoGAN. Please see Fig. 4a for labeling conventions. (b) InfoGAN features and GAN features perform similarly on this dataset. However, supervised learning features are only slightly better than the unsupervised counterparts.

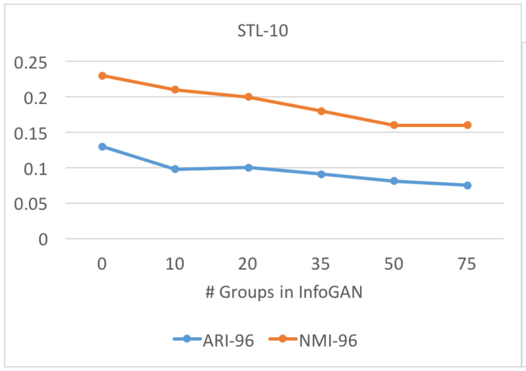
Figure 5b compares the performance when we use different different features. Notice that the features obtained by adversarial training are as competitive as the features obtained using supervised training. We that this is because of two reasons; (i) CIFAR-100 coarse level categories are much harder to distinguish than the CIFAR-10 categories, making it difficult for the supervised model to learn good features, (ii) the number of training examples per category in CIFAR-100 is lesser than CIFAR-10 because we are training using the 20 coarse categories compared with 10 of CIFAR-10. We label the direct prediction results with a (-InfoGAN).

4.3 STL-10

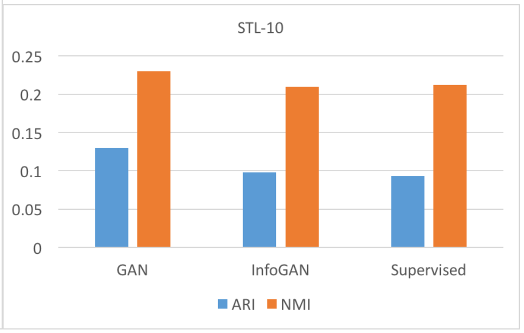
Finally, we also perform experiments on the STL-10 dataset. This database consists of 5000 images for training with labels, 100000 training images without labels, and 8000 images for testing. The dataset consists of 10 categories, and all the images are of size 96×96. This dataset brings out the advantages of unsupervised learning algorithms. The database is more than two times bigger than CIFAR-10 and CIFAR-100 datasets in terms of the number of images and each image is 9 times the size of the CIFAR images. Figure 6b shows that the unsupervised learning with adversarial training outperforms the same models trained using supervised learning. From Figure 6a, we also notice that the features learned using vanilla GAN does better than the features learned using InfoGAN. Increasing the complexity of the datasets makes it difficult for InfoGAN to group the images in the dataset.

5 Conclusion

In this paper, we explore an unsupervised feature learning technique where the model is trained using adversarial training from a generative network. We use a generative model to generate images that act as an adversary to the discriminative network. We explore the standard GAN architecture and the InfoGAN architecture for training the discriminative model. We also show that direct prediction using InfoGAN's recognition model does not always result in identifying object category-level information. Instead, we fuse the features learned by adversarial training with a traditional unsupervised learning approach, k-means clustering, and show that this combination produces better results than direct prediction. We also show that, in situations where there are limited amounts of labeled training data and large amounts of unlabeled data, adversarial training has the potential to outperform supervised learning.



(a)



(b)

Figure 6: STL-10: (a) InfoGAN's performance drops with increase in the number of groups. (b) Vanilla GAN's features outperform InfoGAN-trained features. Also, notice that, with just 5000 labeled training images, supervised learning starts to reach its limits. However, our model makes use of the additional 100000 unlabeled images and is able to learn representations that surpass the performance of features learned using the supervised model.

References

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016a.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*, 2016b.

Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pp. 1486–1494, 2015.

Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *International Conference on Computer Vision*, 2015.

Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 766–774, 2014.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

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.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. 2016. URL `http://arxiv.org/abs/1603.09246`.

Vittal Premachandran, Boyan Bonev, Xiaochen Lian, and Alan L. Yuille. PASCAL boundaries: A class-agnostic semantic boundary dataset. *CoRR*, abs/1511.07951, 2015. URL `http://arxiv.org/abs/1511.07951`.

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

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103. ACM, 2008.

Jianyu Wang, Zhishuai Zhang, Vittal Premachandran, and Alan Yuille. Discovering internal representations from object-cnns using population encoding. *arXiv preprint arXiv:1511.06855*, 2015.

Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2794–2802, 2015.

Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1395–1403, 2015.