Zero-Shot Exploration

Qingshuang Pang

[#]*School of Engineering, The Hong Kong University of Science and Technology*
Hong Kong

[1]`qpang@connect.ust.com`

***Abstract*— Most existing zero-shot learning methods consider the problem as a visual semantic embedding one. Given the demonstrated capability of Generative Adversarial Networks(GANs) to generate images, I instead use GANs to imagine unseen categories from attributes and hence recognize novel classes with no examples being seen. Specifically, I propose a simple generative model that takes as input noisy attributes about an unseen class and generates images for this class. With added data, zero-shot learning is naturally converted to a traditional classification problem. Additionally, to preserve the inter-class discrimination of the generated features, a novel loss function was provided in this structure. I did several experiments of this upgrade GAN for zero shot learning and analyst the potential reasons of the results.**

***Keywords*— Zero-Shot learning, GAN, attributes to image translation**

I. Introduction

Most existing zero-shot learning methods consider the problem as a visual semantic embedding one. Given the demonstrated capability of Generative Adversarial Networks (GANs)[1] to generate images, I instead use GANs to imagine unseen categories from attributes and hence recognize novel classes with no examples being seen. Specifically, I propose a simple generative model that takes as input noisy attributes about an unseen class and generates images for this class. With added data, zero-shot In the conventional object classification tasks, samples of all classes are available for training a model. However, objects in the real world have a long-tailed distribution. In spite of the fact that images of common concepts can be readily found, there remains a tremendous number of concepts with insufficient and sparse visual data, thus making the conventional object classification methods infeasible. Targeting on tackling such an unseen object recognition problem, zero-shot learning has been widely researched recently.

The main idea that ensuring the success of zero-shot learning is to find an intermediate semantic representation (e.g. attributes or textual features) to transfer the knowledge learned from seen classes to unseen ones. The majority of state-of-the-art approaches consider zero-shot learning as a visual-semantic embedding problem. The paradigm can be generalized as training mapping functions that project visual features and/or semantic features to a common embedding space. The class label of an unseen instance is predicted by ranking the similarity scores between semantic features of all unseen classes and the visual feature of the instance in embedding space. Such a strategy conducts a one-to-one projection from semantic space to visual space.

However, attributes descriptions for categories and objects are inherently mapped to a variety of points in the image space. For example, if I gave a three-dimensions attribute: “black”, “water”, “fins”, this could be all marine life in black. This motivates me to study how adversarial training learns a one-to-many mapping with adding information. In this report, I propose a generative adversarial approach for zero-shot learning GAN to generate images that belong to the class that never show on training set on AWA2[2].

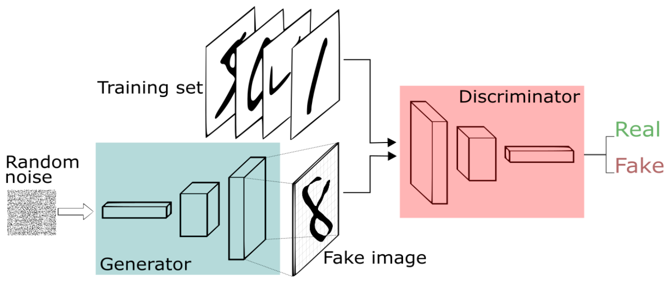
I adopt a novel GAN structure that can transfer attributes and modify them to fit zero-shot learning problems as an image problem. I focus on how to make the structure ensure the intra-class diversity while keeping inter-class discrimination for unseen novel classes. Once this the generator is trained, it can generate images of unseen classes with input real image and unseen attributes,

II. Related Work

In this section, I will give a brief introduction to the related work of my project, which includes GANs, encoder-decoder network and zero-shot learning. This project is mostly based on the fast development of these three areas.

A. Generative Adversarial Networks

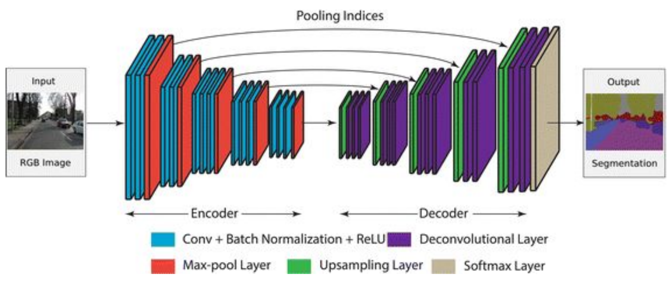
Generative adversarial networks (GAN) is a deep neural network architecture composed of two nets. This is introduced by Ian Goodfellow and other researchers at the University of Montreal. The generative network generates candidates while the discriminative network evaluates them. As the training process goes through, the generative network will generate the information that distributing as similar to the real information as possible so that it will "fool" the discriminative network, which is why it is called adversarial.



As the author introduced in the original paper of GANs, firstly it defines a prior on input noise variables pz(z)To learn the generator's distribution pg over data x.then represent a mapping to data space as G(z; θg), where G is a differentiable function represented by a multilayer perceptron with parameters θg. It also defines a second multilayer perceptron D(x; θd) that outputs a single scalar. When training D, it is going to tell the example is fake or real, ie D(x) is 1 or close to 1 while the example is real and 0 or close to 0 when it is fake. We simultaneously train G to minimize log(1 − D(G(z))). And naturally, we train D to maximize (the loss function of D). By the architecture mentioned above, GANs show a great ability to generate new information that can monitor the real-world information. The key to GANs' success is the idea of the adversarial loss that forces the generated images to be indistinguishable for real photos.

B. Encoder-decoder Networks [3]

An encoder is a network that takes the input and outputs a feature map/vector/tensor. These feature vectors hold the information, the features, that represents the input. The decoder is again a network (usually the same network structure as encoder but in opposite orientation) that takes the feature vector from the encoder and gives the best closest match to the actual input or intended output.



The encoders are trained with the decoders. There are no labels (hence unsupervised). The loss function is based on computing the delta between the actual and reconstructed input. The optimizer will try to train both encoder and decoder to lower this reconstruction loss.

Once trained, the encoder will give a feature vector for input that can be used by decoder to construct the input with the features that matter the most to make the reconstructed input recognizable as the actual input.

In my project, I use the encoder to learn the shape features for animals and add the attribute before the decoder to reconstruct images. It will make the network learning more features at low levels.

C. Zero-shot learning

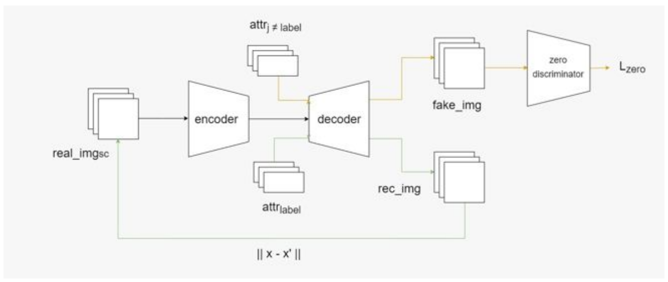
The general question in Zero-shot learning is about: given a semantic encoding of a large set of concept classes, can we build a classifier to recognize classes that were omitted from the training set?

As one of the pioneering works, Lampert et al. proposed a Direct Attribute Prediction (DAP)[4] model that assumed independence of attributes and estimated the posterior of the test class by combining the attribute prediction probabilities. Without the independence assumption, Akata et al. proposed an Attribute Label Embedding(ALE)[5] approach that considers attributes as the semantic embedding of classes and thus tackles ZSL as a visual semantic embedding problem. Consequently, the majority of state-of-the-art methods converge to embedding-based methods. In this project, I use pre-extracted features provided by AWA2 as the embedding space of transfer learning. And it also cause some potential problems that will discuss after the presentation of results.

III. Network Structure

The whole Zero-shot GANs consists of three parts: generator, discriminator and semantic encoder. I explain the structure and layers in the three parts one by one.

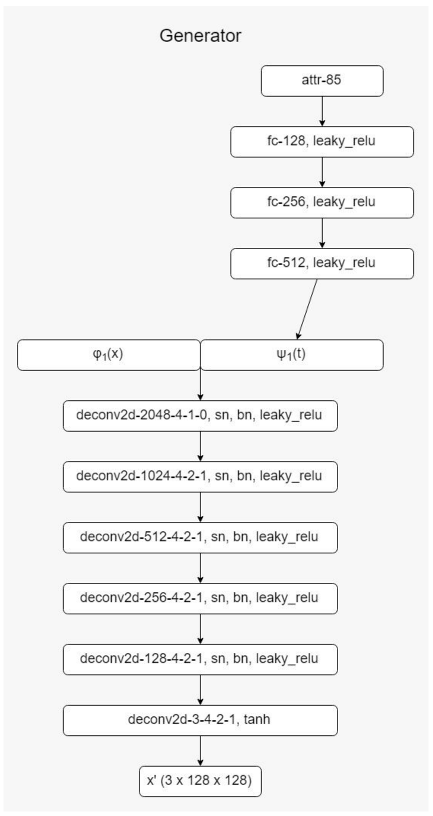
A. Generator



The structure of the generator consists of three convolution layers. Instead of the normal “convolutional layer” - ”batch normalization layer” - ”Relu layer”, I use spectral normalization [6] on the convolutional kernel. This is to solve the unstable problems at the beginning of training. The details will be discussed at 4.1.

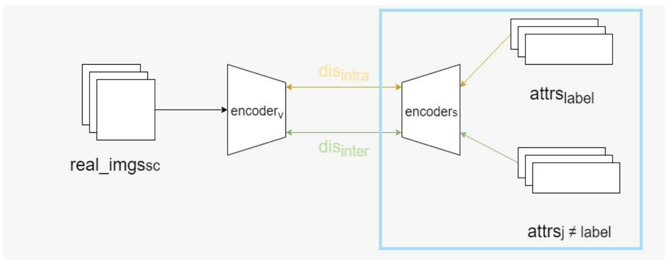
Convolution layers are used to extract lower layer features from the image. After the features were extracted, it would be sent to the decoder network. One was concatenated with the attributes with the same label and the other was concatenated with the attributes with different labels. The first one is the positive sample and the second one is negative for the discriminator.

Finally, deconvolution layers are used to restore an output image, the same size as the input image. This whole process could be viewed as an encoder-decoder.



Generator
attr-85
fc-128, leaky_relu
fc-256, leaky_relu
fc-512, leaky_relu
$\varphi_1(x)$
$\psi_1(t)$
deconv2d-2048-4-1-0, sn, bn, leaky_relu
deconv2d-1024-4-2-1, sn, bn, leaky_relu
deconv2d-512-4-2-1, sn, bn, leaky_relu
deconv2d-256-4-2-1, sn, bn, leaky_relu
deconv2d-128-4-2-1, sn, bn, leaky_relu
deconv2d-3-4-2-1, tanh
x' (3 x 128 x 128)

B. *Discriminator*

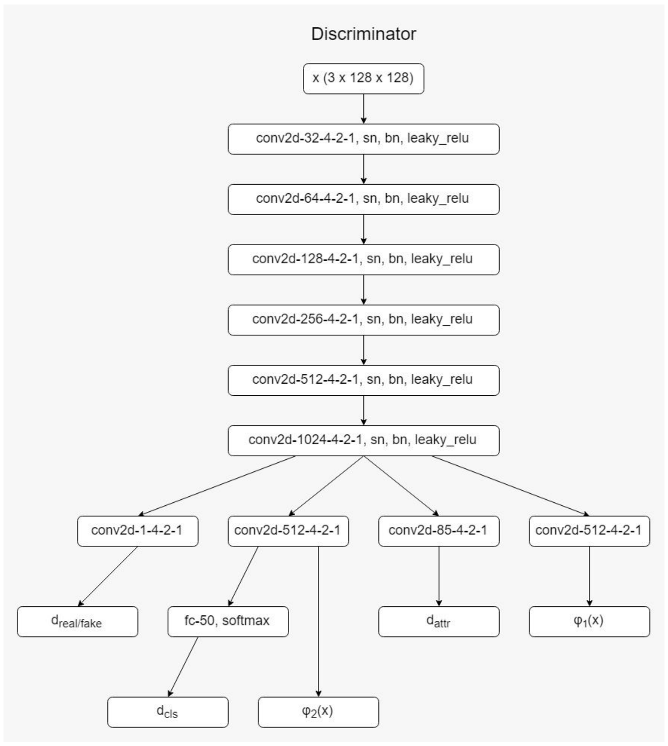


The structure of discriminator is shown as the figure above without the blue frame. The discriminator handles four tasks simultaneously: discriminating whether input images are real or fake, classifying the attributes of the input images, extracting the feature map of the input image as the input of the generator to reconstruct the image and as embedded result comparing with the embedded result of the corresponding attributes.

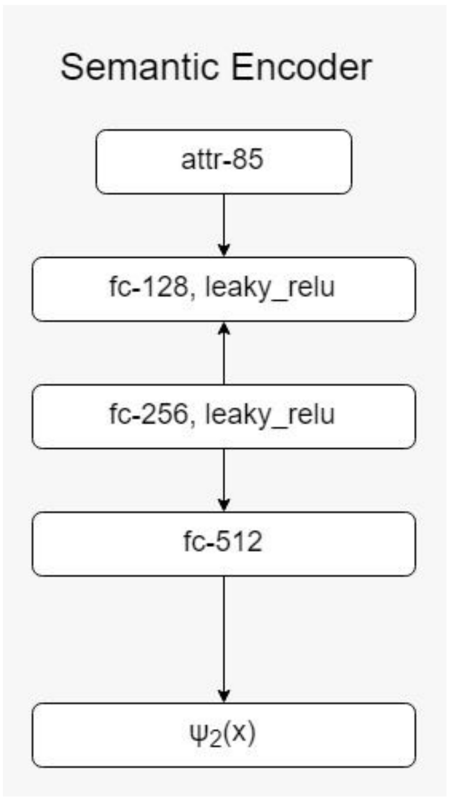
While comparing the embedded result of the image with the embedded result of attributes, the embedded result of the image of the same class tends to be the same. But while reconstructing the image, the model is wanted to give enough diverse results so that the reconstructor would not mix the embedded result of a different image. And therefore, these two tasks do not share the same output layer.

For zero-shot classification, the model would learn a mapping function to bridge the visual space and the semantic space. Embedding images and its attributes to the latent intermediate space is a kind of flexible method because the represented results highlight those attributes which are more visually important. But when bridging then into the intermediate space, when the attributes of all classes are represented by the same latent vector and all the images are represented by the same latent vector, the loss of the model would be minimized to zero. However, in this case, the model is useless. To handle this situation, Zhang et al.[7] propose a method to increase the inter-class separability by connecting a softmax layer after the visual embedding output layer. Also according to their paper, bias issues, which is a common problem existing in some former studies of ZSL, is also solved by their method. The softmax layer is trained only on the images of seen classes.

Also, similar to the structure of AttGAN[8], the discriminator also learns to classify the attributes of images, but this part is only trained on the real images.


Discriminator
x (3 x 128 x 128)
conv2d-32-4-2-1, sn, bn, leaky_relu
conv2d-64-4-2-1, sn, bn, leaky_relu
conv2d-128-4-2-1, sn, bn, leaky_relu
conv2d-256-4-2-1, sn, bn, leaky_relu
conv2d-512-4-2-1, sn, bn, leaky_relu
conv2d-1024-4-2-1, sn, bn, leaky_relu
conv2d-1-4-2-1
conv2d-512-4-2-1
conv2d-85-4-2-1
conv2d-512-4-2-1
dreal/fake
fc-50, softmax
dattr
φ1(x)
dcls
φ2(x)

C. Semantic Encoder



The structure of the semantic encoder is shown at the blue frame of discriminator. The purpose of a semantic encoder is to encode the attributes and using trying to matching the image encoder with the same class. Also diverse the distance of encoded attribute with different classes.

D. Loss Functions

$$L_{D} = L_{D\_real} + L_{D\_fake}$$

The loss function consists of two parts.

$$L_{D\_real} = [1 - cos(\phi(x_i), \psi(t_i))]^2 + \lambda_1 cos(\phi(x_i), \psi(t_j))^2 + \lambda_2 L_{softmax}$$

$$L_{softmax} = \mathbb{E}[logP(C = c|x_{real})]$$

The loss function of discriminator on real images is as shown shown below. φ(x) represent the visual embedded result of images and ψ(t) represents the semantic embedded result of attributes. The optimizer would maximize the cosine similarity of embedded result of real images and the attributes of their corresponding classes, and minimize the cosine similarity of embedded result of real images and the attributes of the other classes. The softmax regular item is to handle the issues mentioned in the former chapter.

$$L_{D\_fake} = cos(\phi(\hat{x}_i), \psi(t_i))^2$$

The loss function of discriminator on fake images is to minimize the cosine similarity in the embedded result of generated images and the embedded result of their attributes.

$$L_G = [1 - cos(\phi(\hat{x}_i), \psi(t_i))]^2$$

And the generator would try to maximize this cosine similarity.

$$L_{D\_rf} = (1 - D_{rf}(x))^2 + D_{rf}(\hat{x})^2$$

$$L_{G\_rf} = (1 - D_{rf}(\hat{x}))^2$$

The adversarial loss is the same as the loss function of LSGAN[9].

$$L_{G\_rec} = ||G(D_{enc}(x)) - x||_1$$

The reconstruction loss function is the L1 distance between the reconstructed images and the original images.

$$L_{D\_attr} = ||D_{attr}(x_i) - t_i||_2$$

$$L_{G\_attr} = ||D_{attr}(\hat{x}_i) - t_i||_2$$

Similar to AttGAN, a classifier would be trained on real images to classify images as their correct attributes. And then this classifier would guide the generator to generate images with the correct attributes.

IV. Training Process

A. Dataset

This dataset provides a platform to benchmark transfer-learning algorithms, in particular, attribute based classification and zero-shot learning. It can act as a drop-in replacement to the original Animals with Attributes (AwA) dataset.

It consists of 37322 images of 50 animals classes with pre-extracted feature representations for each image. The whole dataset is divided into the train set with 40 classes and the test set with 10 classes. The classes are aligned with Osherson's classical class/attribute matrix [3,4], thereby providing 85 numeric attribute values for each class. Using the shared attributes, it is possible to transfer information between different classes.

The pre-extracted features will be used as extra information that providing a latent pattern of different classes. With feeding this information, generators can generator images in unseen classes.

B. Data Preprocessing

· Pretrained Mask R-CNN[10]

Mask R-CNN is an extension over Faster R-CNN. Faster R-CNN predicts bounding boxes and Mask R-CNN essentially adds one more branch for predicting an object mask in parallel. Here I am not going to go on the details of Mask R-CNN but just using it to detect the main object in AWA2 dataset and cropped it for a better performance of Zero-Shot GAN.

I got the pretrained Mask-RCNN model on COCO dataset with ResNet-50[11] backbone for cropping images, and Figure () is an example of images before and after cropping.

·Data augmentation

Upon training, we randomly apply image flopping to artificially increase the complexity of training set, in order to train a more robust model.

After preprocessing, the input shape is set to 128*128*3.

C. Training Details

The main idea to stabilize the GAN training is not to make the discriminator too smart and to add randomness to the training process. Each step of updating the generator is the step it learns how to fool the discriminator. If the discriminator is too smart and therefore cannot be fooled, the generator would not learn anything useful. The randomness is one of the approaches to prevent this situation.

The methods implemented in training are label smoothing and flipping, and adding instance noise to the image before inputting into the discriminator. And the following tricks are also applied during training process.

- Spectral Normalization [6]

Suppose we have a linear function A: $\mathbb{R}^n \rightarrow \mathbb{R}^m$. This function could be the pre-activation operation of one layer in a multilayer perceptron. We can compute the spectral norm of this function, which is defined as the largest singular value of the matrix i.e., the square root of the largest eigenvalue of $A^T A$.

Spectral normalization can ensure the Lipschitz constant of this function, which will reduce the variation of loss during training. The following ensured the correctness of what is shown above. If a function is Lipschitz constant, which means,

$$||Ax|| \leq K||x||$$

for all x∈I. This is equivalent to the statement:

$$\langle Ax, Ax \rangle \leq K^2 \langle x, x \rangle, \forall x \in I$$

which in turn is equivalent to

$$\langle (A^TA - K^2)x, x\rangle \le 0, \forall x \in I.$$

If we expand x in the orthonormal basis of eigenvectors of A^TA (i.e., $x = \sum_i x_i v_i$), we can then write out this inner product explicitly:

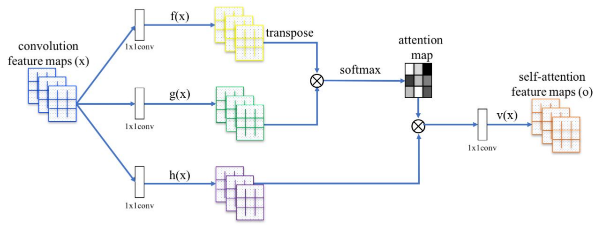
$$\begin{aligned}\langle (A^TA - K^2)x, x\rangle &= \langle (A^TA - K^2)\sum_i x_i v_i, \sum_j x_j v_j\rangle \\ &= \sum_i \sum_j x_i x_j \langle (A^TA - K^2)v_i, v_j\rangle \\ &= \sum_i (\lambda_i - K^2)x_i^2 \le 0 \\ &\implies \sum_i (K^2 - \lambda_i)x_i^2 \ge 0.\end{aligned}$$

Note that since A^TA is positive semidefinite, all the λ_is must be nonnegative. To guarantee the above sum to be nonnegative, each term must be nonnegative, so we have

$$K^2 - \lambda_i \ge 0 \text{ for all } i = 1 \ldots n.$$

Since we choose K to be the minimum value satisfying the above constraints, we immediately see that K is the square root of the largest eigenvalue of A^TA. Therefore, the Lipschitz constant of a linear function is its largest singular value, or its spectral norm.

- Self-Attention GAN [12]



While convolutional filters are good at exploring spatial locality information, the receptive fields may not be large enough to cover larger structures. We can increase the filter size or the depth of the deep network but this will make GANs even harder to train.Alternatively, we can apply the attention concept. The main design is shown above.

We multiple x with Wf and Wg (these are model parameters to be trained) and use them to compute the attention map β with the following formula

$$g(\boldsymbol{x}) = \boldsymbol{W_g x} \qquad \boldsymbol{x} \in \mathbb{R}^{C\times N}, \boldsymbol{W_f} \in \mathbb{R}^{\bar{C}\times C}, \bar{C} = C/8$$

$$f(\boldsymbol{x}) = \boldsymbol{W_f x} \qquad \boldsymbol{W_g} \in \mathbb{R}^{\bar{C}\times C}$$

$$s_{ij} = \boldsymbol{f(x_i)}^T \boldsymbol{g(x_j)}$$

$$\beta_{j,i} = \frac{\exp(s_{ij})}{\sum_{i=1}^{N} \exp(s_{ij})} \qquad \beta \in \mathbb{R}^{N\times N}$$

$\beta_{j,i}$ indicates the extent to which the model attends to the i^{th} location when synthesizing the j^{th} region.

For each spatial location, an attention map is created which acts as a mask. βij is interpreted as the impact of location i when rendering the location j.

- TTUR (Two Time-scale Update Rule)[13]

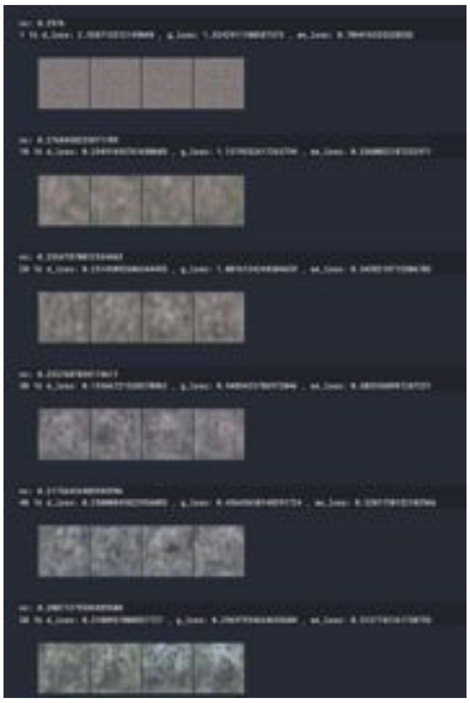
V. S. Borkar proposes a two time-scale update rule (TTUR) for GAN training in “Stochastic approximation with two time scales”, which is to use different learning rates for the discriminator and the generator. Under certain assumptions, TTUR allows the proof of convergence of GANs, and the results carry to Adam. To evaluate the effectiveness of TTUR, the paper additionally proposed Frechet Inception Distance (FID), i.e., the approximate Frechet distance in the code space defined by the Inception model. Empirically, by tuning the two learning rates carefully, TTUR is able to achieve faster and more stable training as measured by FID, with comparable sample quality given the same FID score.

- Parameters setting

During the experiment, I find that TTUR can optimize the performance of GAN effectively with the learning rate of generator set as 5e-5 and the learning rate of discriminator set as 2e-4 as the initial learning rate, and linearly reduce it to 0 after 300 epochs, during another 300-epoch period of time.

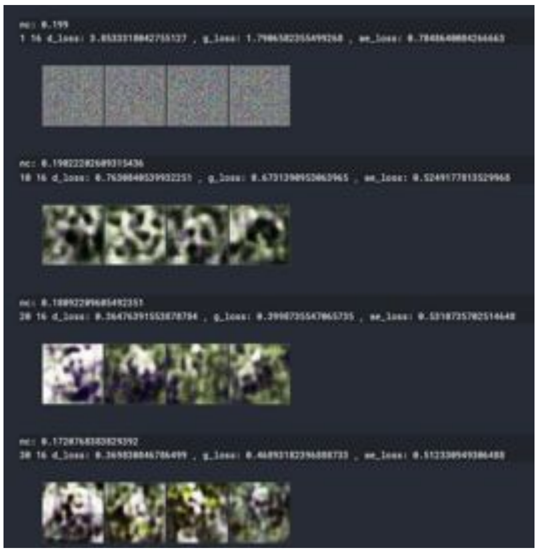
In addition to the softmax function, another regular item implemented in the loss function is the cosine similarity between the visual embedded result of the image and the semantic embedded result of the attributes of the other classes. The optimizer also tries to minimize this regular item but with lower priority than to maximize the cosine similarity between the embedded result of the image and the embedded result of its corresponding attributes. And the λ1 is the parameter to control the priority of this regular item

With the same parameters' setting, when the layers are norma convolution layer and no other optimization, the first epchs training result is as shown below with no signal of converging.



```
ac: 0.2[illegible]
1 16 d_loss: [illegible] , g_loss: [illegible] , ae_loss: [illegible]

ac: [illegible]
10 16 d_loss: [illegible] , g_loss: [illegible] , ae_loss: [illegible]

ac: [illegible]
20 16 d_loss: [illegible] , g_loss: [illegible] , ae_loss: [illegible]

ac: [illegible]
30 16 d_loss: [illegible] , g_loss: [illegible] , ae_loss: [illegible]

ac: [illegible]
40 16 d_loss: [illegible] , g_loss: [illegible] , ae_loss: [illegible]

ac: [illegible]
50 16 d_loss: [illegible] , g_loss: [illegible] , ae_loss: [illegible]
```

However, after the GAN training trick it gets a much larger rate of converge.



```
ac: 0.199
1 16 d_loss: 3.0533318042755127 , g_loss: 1.7906582355499268 , ae_loss: 0.7848640084266663

ac: 0.19822202609315436
10 16 d_loss: 0.7630040539932251 , g_loss: 0.6731390953063965 , ae_loss: 0.5249177813529968

ac: 0.18092209605492351
20 16 d_loss: 0.3647629155387879 , g_loss: 0.3998735547065735 , ae_loss: 0.5318735702514648

ac: 0.17287683828293592
30 16 d_loss: 0.3693004467064 99 , g_loss: 0.4689318229688733 , ae_loss: 0.5123309493064 88
```

V. Test

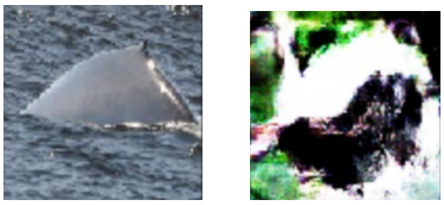
A. Test result

Because the awa2 dataset is limited and diverse, the training process is hard to stabilize and the quality of generated image is poor. But the model still can realize the environment transfer of the image.

the transferred image has even features to be distinguished as a skunk, but it lacks details.

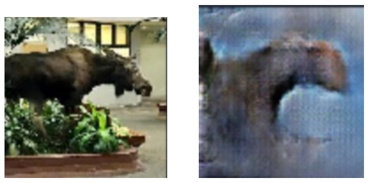
For the unseen class, the generated result is even not distinguishable but it is able to see that the background of the generated result is blue, the color of the sea. And the generated result of leopard also show the color of grass.

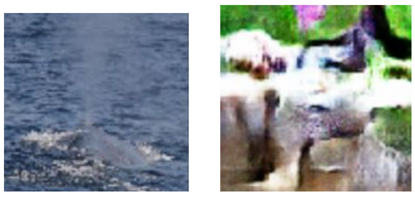
the image translate from seen class to seen class



blue whale to skunk

the image translate from seen class to unseen class


moose to humpback whale


blue whale to leopard

B. Analysis

It is obvious that this GANs only learned the limited dimensions of features such as texture, color, and environment. And missed features include structure, shape, position and etc. This is giving me a hit that if we can use another neural network with detection method to generate higher dimensions attributes.

VI. Conclusion

During this project, I first do a survey of zero-shot learning to get a basic understanding of this area. Then I propose a GANs structure to produce the images from unseen classes. By modifying the discriminator, it can be transferred to a zero-shot classification problems. Then I build the Zero-Shot GANs from sketch via Pytorch then do the test about the performance of this GANs. Though it does not show a very good performance, it would be a different thought of GANs and ZSL fusion. Also for any algorithm, it can not get the training set that includes everything in the real world. So zero-shot learning will have more potential than it is already showing now,

*https://github.com/pqingshuang/zero-shot-learning/blob/master/zero-15-attn.ipynb The source code is posted here.

References

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672-2680).

[2] Y. Xian, C. H. Lampert, B. Schiele, Z. Akata. *"Zero-Shot Learning - A Comprehensive Evaluation of the Good, the Bad and the Ugly"*, IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) 40(8), 2018. (arXiv:1707.00600 [cs.CV])

[3] Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1125-1134).

[4] Lampert, C. H., Nickisch, H., & Harmeling, S. (2013). Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(3), 453-465.

[5] Akata, Z., Perronnin, F., Harchaoui, Z., & Schmid, C. (2015). Label-embedding for image classification. *IEEE transactions on pattern analysis and machine intelligence*, *38*(7), 1425-1438.

[6] Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*.

[7] Zhang, L., Wang, P., Liu, L., Sehn, C. (2018). Towards Effective Deep Embedding for Zero-Shot Learning. *arXiv preprint arXiv:1808.10075v2*

[8] He, Z., Zuo, W., Kan, M., Shan, S., & Chen, X. (2019). Attgan: Facial attribute editing by only

changing what you want. IEEE Transactions on Image Processing.

[9] Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., & Paul Smolley, S. (2017). Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2794-2802).

[10] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 2961-2969).

[11] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[12] Zhang, H., Goodfellow, I., Metaxas, D., & Odena, A. (2018). Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*..

[13] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems* (pp. 6626-6637).