Evaluation of Generative Models: *Practice*

Hao Dong

Peking University

Evaluation of Generative Models: Practice

- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

DCGAN Evaluation: Classification Accuracy

- Given pretrained DCGAN for MNIST, how to evaluate it?

- Classification accuracy
  - Use the discriminator's convolutional features from all layers
  - Maxpooling each layers representation to produce a 4 × 4 spatial grid
  - Flatten and concatenate these features to form a 28672 dimensional vector
  - A regularized linear L2-SVM classifier is trained on top of them

Table 1: CIFAR-10 classification results using our pre-trained model. Our DCGAN is not pre-trained on CIFAR-10, but on Imagenet-1k, and the features are used to classify CIFAR-10 images.

Model	Accuracy	Accuracy (400 per class)	max # of features units
1 Layer K-means	80.6%	63.7% (±0.7%)	4800
3 Layer K-means Learned RF	82.0%	70.7% (±0.7%)	3200
View Invariant K-means	81.9%	72.6% (±0.7%)	6400
Exemplar CNN	84.3%	77.4% (±0.2%)	1024
DCGAN (ours) + L2-SVM	82.8%	73.8% (±0.4%)	512

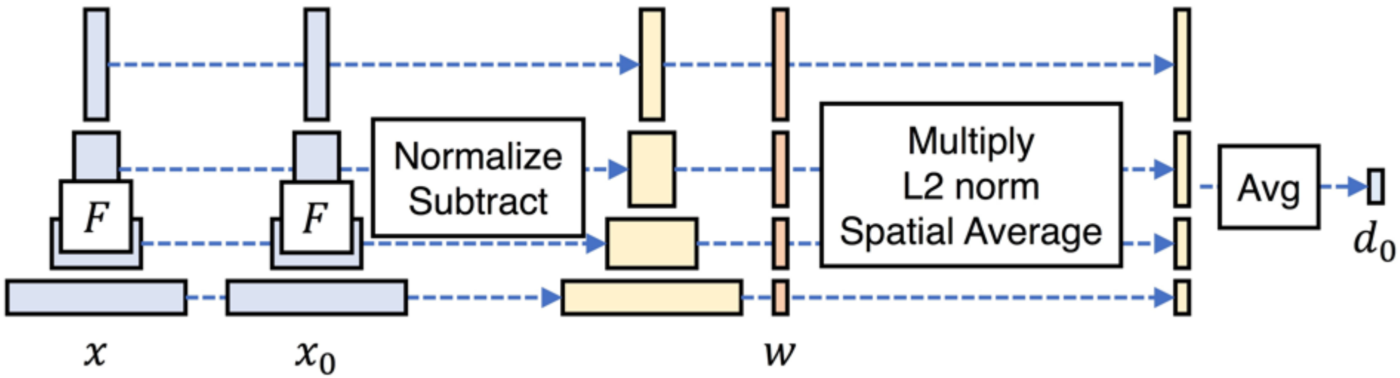
- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

DCGAN Evaluation: LPIPS

- Given pretrained DCGAN for MNIST, how to evaluate it?

- **Learned Perceptual Image Patch Similarity (LPIPS)**
  - To evaluate the diversity of the generation
  - Perceptual similarity is an emergent property shared across deep visual representations.

Try the LPIPS Metric/Download the Dataset



- **Implementation:** https://github.com/richzhang/PerceptualSimilarity

- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

VAE Evaluation

- Given pretrained VAE models for MNIST, how to evaluate it?
- **Negative Log Likelihood (NLL)**
  - NLL represents the probability of generating real data
  - Less NLL indicated better generation of VAE

- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

VAE Evaluation

- Given pretrained VAE models for MNIST, how to evaluate it?

- **Beta-VAE metric and Mutual Information Gap (MIG)**
  - To evaluate the disentanglement of VAE.
  - Beta-VAE metric is the accuracy of a linear classifier that predicts a fixed factor of variation
  - MIG is the gap between the largest and second largest mutual information
  - Review lecture 20 for more details

beta_vae.py	internal change	5 months ago
beta_vae_test.py	internal change	5 months ago
mig.py	internal change	5 months ago
mig_test.py	internal change	5 months ago

- **ICML 2019 Best Paper**
- Implementation:

https://github.com/google-research/disentanglement_lib/tree/master/disentanglement_lib/evaluation/metrics

- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

VAE Evaluation: Clustering

- Given pretrained VAE for MNIST, how to evaluate it?

- Clustering
  - **Completeness score** (between [0, 1])
  - **Homogeneity score** (between [0, 1])
  - **V measure score** (also called normalized mutual information, between [0, 1])

$$c = 1 - \frac{H(K|C)}{H(K)} \qquad h = 1 - \frac{H(C|K)}{H(C)} \qquad v = 2 \cdot \frac{h \cdot c}{h + c}$$

- Review lecture 20's slides for the implementation

- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

Model Size

- The size of model is also an important metric of generative models
  - The size is the number of parameters of the model
  - It indicates the scalability of the model
  - Less parameters required, stronger scalability of the model

- Example: StarGAN Evaluation

Method	Classification error	# of parameters
DIAT	4.10	52.6M $\times$ 7
CycleGAN	5.99	52.6M $\times$ 14
IcGAN	8.07	67.8M $\times$ 1
StarGAN	**2.12**	**53.2M** $\times$ **1**
Real images	0.45	-

Table 3. Classification errors [%] and the number of parameters on the RaFD dataset.

  - The smallest size of StarGAN indicated its advantage in multi-domain translation

- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

Implementation: tensorlayer Model.weight

- Calculate the size of a model using tensorlayer is convenient
- NLL is a term in the loss of classical VAE

```python
def count_weights(model):
    n_weights = 0
    for i, w in enumerate(model.all_weights):
        n = 1
        # for s in p.eval().shape:
        for s in w.get_shape():
            try:
                s = int(s)
            except:
                s = 1
            if s:
                n = n * s
        n_weights = n_weights + n
    print("num of weights (parameters) %d" % n_weights)
    return n_weights
```

```python
if __name__ == '__main__':
    Ea = get_Ea()
    Ea.eval()
    print("Ea:")
    count_weights(Ea)

    Ec = get_Ec()
    Ec.eval()
    print("Ec:")
    count_weights(Ec)

    D = get_D()
    D.eval()
    print("D:")
    count_weights(D)

    G = get_G()
    G.eval()
    print("G:")
    count_weights(G)
```

- Try to evaluate the size of DCGAN and VAE by yourself!

Summary

- DCGAN Evaluation
  - Classification accuracy
  - LPIPS
- VAE Evaluation
  - NLL
  - Beta-VAE metric
  - MIG
  - Clustering
- Others
  - Model Size
  - Tensorlayer Model.weights

Thanks