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 $4 \times 4$ 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.

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

• **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

- **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])

\[
c = 1 - \frac{H(K|C)}{H(K)} \quad h = 1 - \frac{H(C|K)}{H(C)} \quad v = 2 \cdot \frac{h \cdot c}{h + c}
\]

• 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

<table>
<thead>
<tr>
<th>Method</th>
<th>Classification error</th>
<th># of parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIAT</td>
<td>4.10</td>
<td>52.6M × 7</td>
</tr>
<tr>
<td>CycleGAN</td>
<td>5.99</td>
<td>52.6M × 14</td>
</tr>
<tr>
<td>IcGAN</td>
<td>8.07</td>
<td>67.8M × 1</td>
</tr>
<tr>
<td>StarGAN</td>
<td>2.12</td>
<td><strong>53.2M × 1</strong></td>
</tr>
<tr>
<td>Real images</td>
<td>0.45</td>
<td>-</td>
</tr>
</tbody>
</table>

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(f'num of weights (parameters) %d' % n_weights)
    return n_weights
```

- 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