Deep Hyperspherical Learning

Weiyang Liu, Yan-Ming Zhang, Xingguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, Le Song

Georgia Institute of Technology
Motivation

• 2D Fourier Transform for images

\[ \text{The magnitude of } A + \text{The phase of } B = \]

\[ \text{The magnitude of } B + \text{The phase of } A = \]

• Phase contains the crucial discriminative information!
**SphereNet**: a network that focuses on the angular (phase) information

- Hyperspherical Convolutional (SphereConv) Operator:

  \[ \mathcal{F}_s(w, x) = g(\theta(w, x)) + b \mathcal{F}_s \]

  Where \( \theta(w, x) \) is the angle between the kernel parameter \( w \) and the local patch \( x \). A simple example is cosine SphereConv:

  \[ g(\theta(w, x)) = \cos(\theta(w, x)) \]

- We use this SphereConv operator to replace the original inner product based convolutional operator in the CNNs, and propose the **SphereNet**.  (SphereNet comes from that angle can be viewed as the geodesic distance on a unit hypersphere)
Four SphereConv operators

- **linear SphereConv**
  \[ g(\theta_{(w,x)}) = a\theta_{(w,x)} + b \]

- **cosine SphereConv**
  \[ g(\theta_{(w,x)}) = \cos(\theta_{(w,x)}) \]

- **sigmoid SphereConv**
  \[ g(\theta_{(w,x)}) = \frac{1 + \exp(-\frac{\pi}{2k})}{1 - \exp(-\frac{\pi}{2k})} \times \frac{1 - \exp\left(\frac{\theta_{(w,x)}}{k} - \frac{\pi}{2k}\right)}{1 + \exp\left(\frac{\theta_{(w,x)}}{k} - \frac{\pi}{2k}\right)} \]

- **Learnable SphereConv**
  \[ g(\theta_{(w,x)}) = \frac{1 + \exp(-\frac{\pi}{2k})}{1 - \exp(-\frac{\pi}{2k})} \times \frac{1 - \exp\left(\frac{\theta_{(w,x)}}{k} - \frac{\pi}{2k}\right)}{1 + \exp\left(\frac{\theta_{(w,x)}}{k} - \frac{\pi}{2k}\right)} \]  
  with the parameter \( k \) to be learned in back-prop
Theoretical Insights

- Suppose the observation is $F = U^*V^*$ (ignore the bias), where $U^* \in \mathbb{R}^{n \times k}$ is the weight, $V^* \in \mathbb{R}^{m \times k}$ is the input that embeds weights from previous layers.

Scaling issue of neural networks:

- Consider the objective:
  \[ \min_{U \in \mathbb{R}^{n \times k}, V \in \mathbb{R}^{m \times k}} G(U, V) = \frac{1}{2} \| F - UV^T \|_F^2. \]

- **Lemma1**: Consider a pair of global optimal points $U, V$ satisfying $F = UV^T$ and $\text{Tr}(V^TV \otimes I_n) \leq \text{Tr}(U^TU \otimes I_m)$. For any real $c > 1$, let $\tilde{U} = cU$ and $\tilde{V} = V/c$, then we have $\kappa(\nabla^2 G(\tilde{U}, \tilde{V})) = \Omega(c^2 \kappa(\nabla^2 G(U, V)))$, where $\kappa = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}}$ is the restricted condition number with $\lambda_{\text{max}}$ being the largest and $\lambda_{\text{min}}$ being the smallest nonzero eigenvalues.

Insensitiveness to Scaling for SphereConv:

- Consider our proposed cosine SphereConv operator, an equivalent problem is:
  \[ \min_{U \in \mathbb{R}^{n \times k}, V \in \mathbb{R}^{m \times k}} G_S(U, V) = \frac{1}{2} \| F - DUUV^TDV \|_F^2 \]
  where $D_U = \text{diag}(\frac{1}{\|U_1\|_2}, \ldots, \frac{1}{\|U_n\|_2}) \in \mathbb{R}^{n \times n}$ and $D_V = \text{diag}(\frac{1}{\|V_1\|_2}, \ldots, \frac{1}{\|V_m\|_2}) \in \mathbb{R}^{m \times m}$ are diagonal matrices.

- **Lemma2**: For any real $c > 1$, let $\tilde{U} = cU$ and $\tilde{V} = V/c$, then we have $\lambda_i(\nabla^2 G_S(\tilde{U}, \tilde{V})) = \lambda_i(\nabla^2 G_S(U, V))$ for all $i \in [(n + m)k] = \{1, 2, \ldots, (n + m)k\}$ and $\kappa(\nabla^2 G(\tilde{U}, \tilde{V})) = \kappa(\nabla^2 G(U, V))$, where $\kappa$ is defined as in Lemma1.
• Regular Neural Nets: scales as $\Omega(c^2)$
• SphereConv: insensitive to scaling
More on SphereNets

• SphereConv can also be used to the fully connected layers, recurrent layers, etc.

• SphereConv can also be viewed as a normalization method that could avoid covariate shift (due to the bounded outputs), and can work simultaneously with Batch Normalization.

• We also design angular loss functions for SphereConv, i.e., generalized angular softmax (GA-Softmax) loss

\[ L_i = -\log \left( \frac{e^\|x_i\| g(m\theta_{y_i,i})}{e^\|x_i\| g(m\theta_{y_i,i}) + \sum_{j \neq y_i} e^\|x_i\| g(\theta_{j,i})} \right) \]
Experiments

- Faster Convergence and better accuracy on CIFAR-10, CIFAR-100

(a) ResNet vs. SphereResNet on CIFAR-10/10+

(b) CNN vs. SphereNet (orth.) on CIFAR-10
Experiments

- SphereConv can be used as a new normalization method (SphereNorm), comparable to Batch Normalization. But they can be used simultaneously.

- The advantages of SphereNorm are very significant, especially with small mini-batch size.

Mini-batch size = 4!
Experiments

• Faster Convergence and comparable accuracy on Imagenet-2012
Visualization on MNIST

Original CNN

SphereNet
More experiments

• Using the SphereConv only to the last fully connected layer gives impressive results on face recognition.

<table>
<thead>
<tr>
<th>Method</th>
<th>protocol</th>
<th>Rank1 Acc.</th>
<th>Ver.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTechLAB - facenx large</td>
<td>Large</td>
<td>73.300</td>
<td>85.081</td>
</tr>
<tr>
<td>Vocord - DeepVo1</td>
<td>Large</td>
<td>75.127</td>
<td>67.318</td>
</tr>
<tr>
<td>Deepsense - Large</td>
<td>Large</td>
<td>74.799</td>
<td>87.764</td>
</tr>
<tr>
<td>Shanghai Tech</td>
<td>Large</td>
<td>74.049</td>
<td>86.369</td>
</tr>
<tr>
<td>Google - FaceNet v8</td>
<td>Large</td>
<td>70.496</td>
<td>86.473</td>
</tr>
<tr>
<td>Beijing FaceAll_Norm_1600</td>
<td>Large</td>
<td>64.804</td>
<td>67.118</td>
</tr>
<tr>
<td>Beijing FaceAll_1600</td>
<td>Large</td>
<td>63.977</td>
<td>63.960</td>
</tr>
<tr>
<td>Deepsense - Small</td>
<td>Small</td>
<td>70.983</td>
<td>82.851</td>
</tr>
<tr>
<td>SIAT_MMLLAB</td>
<td>Small</td>
<td>65.233</td>
<td>76.720</td>
</tr>
<tr>
<td>Barebones FR - cnn</td>
<td>Small</td>
<td>59.363</td>
<td>59.036</td>
</tr>
<tr>
<td>NTechLAB - facenx_small</td>
<td>Small</td>
<td>58.218</td>
<td>66.366</td>
</tr>
<tr>
<td>3DiVi Company - tdvm6</td>
<td>Small</td>
<td>33.705</td>
<td>36.927</td>
</tr>
<tr>
<td>Softmax Loss</td>
<td>Small</td>
<td>54.855</td>
<td>65.925</td>
</tr>
<tr>
<td>Softmax+Contrastive Loss [26]</td>
<td>Small</td>
<td>65.219</td>
<td>78.865</td>
</tr>
<tr>
<td>Triplet Loss [22]</td>
<td>Small</td>
<td>64.797</td>
<td>78.322</td>
</tr>
<tr>
<td>L-Softmax Loss [16]</td>
<td>Small</td>
<td>67.128</td>
<td>80.423</td>
</tr>
<tr>
<td>Softmax+Center Loss [34]</td>
<td>Small</td>
<td>65.494</td>
<td>80.146</td>
</tr>
<tr>
<td>SphereFace (single model)</td>
<td>Small</td>
<td>72.729</td>
<td>85.561</td>
</tr>
<tr>
<td>SphereFace (3-patch ensemble)</td>
<td>Small</td>
<td>75.766</td>
<td>89.142</td>
</tr>
</tbody>
</table>
The End

• The code will be made available at https://github.com/wy1iu/SphereNet
• The code of SphereFace is available at https://github.com/wy1iu/sphereface