Singular Value Decomposition

- SVD constitutes a bridge between Linear Algebra and Multi-Layer Neural Networks.
- SVD results in a good initial guess for weight and bias parameters.
It is a decomposition of an arbitrary matrix $A$ of size $m \times n$ into three factors:

$$A = USV^T$$

where $U$ and $V$ are orthonormal and $S$ is of identical size as $A$, consisting of a diagonal matrix $D$ and a zero matrix.

SVD is then simplified to:

$$A = USV^T = U[S_0, 0][V_0, V_s]^T = US_0V_0^T$$

Motivation

A further application of SVD is an explicit formula for a matrix pseudo-inverse. Pseudo-inverse $A^+$ is the analog of an inverse matrix for the case of non-square matrices, with the property

$$AA^+A = A$$

It can be easily computed with the help of SVD:

$$X^\perp = VS_0^T, U^T$$

Over-Determined: For over-determined problems with input $X$ and output $Y$, the least-square-optimum solution is the linear regression with matrix $B$:

$$y = Bx + a$$

Under-Determined: For under-determined problems, like computer vision or corpus-based semantics, the number of training examples is substantially lower than the dimensions of the input. The projection operation for input $X$ and output $Y$ is then given as:

$$\hat{x} = XX^+x = X(X^TX)^{-1}X^Ty$$

SVD and Linear Networks

Linear Networks with one hidden layer of size $p$ can represent the best linear mapping from input $x$ to output $y$: $y = Bx$
The best approximation with rank limitation to $\hat{r}$ is:

$$y = U_rS_rV_r^T x$$

SVD and Initializing Nonlinear Neural Networks

The nonlinear activation function $\text{sigmoid}$

$$s(x) = \frac{1}{1 + e^{-x}}$$
is nearly linear around $x = 0$ with its derivative $0.25$. By rescaling this unit to

$$f(x) = 2s(x) - 1 = \frac{2}{1 + e^{-2x}} - 1$$
it becomes a nonlinear function which is unity around $x = 0$.

- A Neural Network with one hidden layer using $\text{sigmoid}$ behaves like a linear network for small activation values of the hidden layer.

Computing Experiments

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Init.</th>
<th>size class A</th>
<th>#input</th>
<th>#output</th>
<th>#hidden</th>
<th>#training</th>
<th>#parameters</th>
<th>#constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVD</td>
<td>—</td>
<td>—</td>
<td>10.200</td>
<td>—</td>
<td>10.599</td>
<td>—</td>
<td>10.814</td>
<td>—</td>
</tr>
<tr>
<td>SGD</td>
<td>Random</td>
<td>200</td>
<td>30.361</td>
<td>2000</td>
<td>90.402</td>
<td>4000</td>
<td>177.095</td>
<td>—</td>
</tr>
<tr>
<td>RMSprop</td>
<td>Random</td>
<td>200</td>
<td>0.040</td>
<td>2000</td>
<td>0.096</td>
<td>2000</td>
<td>0.260</td>
<td>—</td>
</tr>
<tr>
<td>CG</td>
<td>Random</td>
<td>637</td>
<td>0.002</td>
<td>821</td>
<td>0.012</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>RMSprop</td>
<td>SVD</td>
<td>2000</td>
<td>0.030</td>
<td>2000</td>
<td>0.085</td>
<td>2000</td>
<td>0.248</td>
<td>—</td>
</tr>
<tr>
<td>Adadelta</td>
<td>SVD</td>
<td>2000</td>
<td>0.062</td>
<td>2000</td>
<td>0.511</td>
<td>2000</td>
<td>2.086</td>
<td>—</td>
</tr>
<tr>
<td>CG</td>
<td>SVD</td>
<td>316</td>
<td>0.000</td>
<td>233</td>
<td>0.021</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

Conclusion

SVD can be used as a good initial guess for the network parameters. The quality of this initial guess may be better than weakly performing (but widely used) methods such as SGD ever reach.

Future Work

- Apply in large datasets (MNIST, CIFAR, etc.)
- Directly implement in TensorFlow: Custom initialization using SVD and Conjugate Gradient Optimization

Bernhard Bermeitinger 1,2 • Tomas Hrycej 1 • Siegfried Handschuh 1,2
{bernhard.bermeitinger,tomas.hrycej,siegfried.handschuh}@unisg.ch

1 Institute for Computer Science, Universität St.Gallen, Switzerland 2 FIM, Universität Passau, Germany

Theoretical Neural Computation, Information and Optimization, Deep Learning

ICANN 2019
28th International Conference on Artificial Neural Networks
17.–19. September 2019, Munich, Germany

# 77