Greedy Deep Transform Learning

Authors: Jyoti Maggu, Angshul Majumdar
Email: jyotim@iiitd.ac.in
Contents

- Introduction
- Deep Representation Learning
  - Stacked AutoEncoder
  - Deep Belief Network
  - Deep Dictionary Learning
- Transform Learning
- Problem Statement
- Proposed Technique
- Experimental Results
- Conclusion
- Future work
• A new tool for Deep Learning.
• Stack one transform after another.
• Learning is done in greedy fashion.
Deep Representation Learning

1. Stacked AutoEncoder
Deep Representation Learning

1. Stacked AutoEncoder

- Nesting one AE inside another.
- Solved using greedy paradigm.
- Used generally for classification.
Deep Representation Learning

1. Stacked AutoEncoder
2. Deep Belief Network
   ○ Undirected Graph Model.
   ○ Information content is preserved by cosine similarity between projection of data and learnt features.
   ○ Probabilistic Formulation.

Restricted Boltzmann Machine
Deep Representation Learning

1. Stacked AutoEncoder
2. Deep Belief Network

Deep Belief Network: Stacking one RBM into another.
Deep Representation Learning

1. Stacked AutoEncoder
2. Deep Belief Network
3. Deep Dictionary Learning

Dictionary Learning

- It learns basis for representing data.
- Columns of dictionary (Atoms) are connections between input and representation layer.
- \[ X = D Z \]
Deep Representation Learning

1. Stacked AutoEncoder
2. Deep Belief Network
3. Deep Dictionary Learning

Deep Dictionary Learning

- Stacking one layer after another.
- $Z_1 = D_2 Z_2$
- $X = D_1 Z_1; \quad X = D_1 D_2 Z_2$
Deep Representation Learning

1. Stacked AutoEncoder
2. Deep Belief Network
3. Deep Dictionary Learning
4. Deep Transform Learning

\[
\min_{T,Z} ||TX-Z||^2_F + \mu ||Z||_0
\]

Transform Learning

- While Dictionary Learning is a synthesis formulation, Transform Learning is its analysis equivalent.
- It learns a transform \(T\) such that it operates on the data \(X\) to generate the coefficients \(Z\).
Deep Representation Learning

1. Stacked AutoEncoder
2. Deep Belief Network
3. Deep Dictionary Learning
4. Deep Transform Learning
Problem Statement

- Given data from different classes; classify them accurately.
- Compare with existing techniques: SAE, DBN, DDL.
- Reduce train time and test feature generation time.
Solution: Transform Learning

\[
\min_{T,Z} \|TX-Z\|_F^2 + \mu \|Z\|_0
\]

- But this leads to degenerate solution; trivial would be \(T=0, Z=0\).
- So, to avoid this; new formulation becomes:

\[
\min_{T,Z} \|TX-Z\|_F^2 + \mu \|Z\|_1 + \lambda (\varepsilon \ |\|T\|_F^2 - \log\det T)
\]

- This can be solved by alternating minimization iteratively.

\[Z \leftarrow \min_Z \|TX-Z\|_F^2 + \mu \|Z\|_o\]

\[T \leftarrow \min_T \|TX-Z\|_F^2 + \lambda (\varepsilon \ |\|T\|_F^2 - \log\det T)\]
Greedy Deep Transform Learning

- Deeper representations are learnt by stacking one transform after another.
- The learning is done in a greedy fashion.

\[ T_N(\phi... (T_2(\phi(T_1X)))) = Z \]

- By substituting, \( T_{N-1}Z_{N-2} = \phi^{-1}(Z_{N-1}) \) and so on, till

\[ T_1X = \phi^{-1}(Z_1) \]

\[ \min_{T_1Z_1} ||T_1X - Z_1||_F^2 + \lambda(||T_1||_F^2 - \log \det T_1) \]

\[ Z_1 = T_1X \]

\[ \min_{T_2Z_2} ||T_2Z_1 - Z_2||_F^2 + \lambda(||T_2||_F^2 - \log \det T_2) \]
Experimental Results:

1. Accuracy improves by using Deep architectures.

Datasets used: MNIST, CIFAR-10 and SVHN

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>97.27</td>
<td>97.66</td>
<td>97.94</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>81.12</td>
<td>81.89</td>
<td>82.60</td>
</tr>
<tr>
<td>SVHN</td>
<td>91.97</td>
<td>92.68</td>
<td>93.00</td>
</tr>
</tbody>
</table>
Experimental Results:

1. Accuracy improves by using Deep architectures.
2. Results with Nearest Neighbours:

Datasets used: MNIST, CIFAR-10 and SVHN

Results with NN classifier using features from DTL:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Proposed</th>
<th>SDAE</th>
<th>DBN</th>
<th>DDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>97.94</td>
<td>97.33</td>
<td>97.05</td>
<td>97.75</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>82.60</td>
<td>78.62</td>
<td>73.96</td>
<td>81.09</td>
</tr>
<tr>
<td>SVHN</td>
<td>93.00</td>
<td>91.11</td>
<td>88.29</td>
<td>92.26</td>
</tr>
</tbody>
</table>
Experimental Results:

1. Accuracy improves by using Deep architectures.

2. Results with Nearest Neighbours:

3. Results with SRC:

Datasets used: MNIST, CIFAR-10 and SVHN

Results with SRC classifier using features from DTL:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Proposed</th>
<th>SDAE</th>
<th>DBN</th>
<th>DDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>98.96</td>
<td>98.33</td>
<td>98.43</td>
<td>98.81</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>85.06</td>
<td>79.32</td>
<td>75.02</td>
<td>83.75</td>
</tr>
<tr>
<td>SVHN</td>
<td>94.55</td>
<td>92.05</td>
<td>90.11</td>
<td>93.62</td>
</tr>
</tbody>
</table>
Experimental Results:

1. Accuracy improves by using Deep architectures.
2. Results with Nearest Neighbours:
3. Results with SRC:
4. Results with SVM:

Datasets used: MNIST, CIFAR-10 and SVHN

Results with SVM classifier using features from DTL:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Proposed</th>
<th>SDAE</th>
<th>DBN</th>
<th>DDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>98.94</td>
<td>97.05</td>
<td>98.44</td>
<td>98.64</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>85.55</td>
<td>78.90</td>
<td>74.30</td>
<td>84.96</td>
</tr>
<tr>
<td>SVHN</td>
<td>95.42</td>
<td>92.60</td>
<td>89.70</td>
<td>93.81</td>
</tr>
</tbody>
</table>
Experimental Results:

1. Accuracy improves by using Deep architectures.
2. Results with Nearest Neighbours:
3. Results with SRC:
4. Results with SVM:
5. Feature generation time:

Time in seconds:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Proposed</th>
<th>SDAE</th>
<th>DBN</th>
<th>DDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>25</td>
<td>120408</td>
<td>30071</td>
<td>107</td>
</tr>
<tr>
<td>Testing</td>
<td>50</td>
<td>61</td>
<td>50</td>
<td>79</td>
</tr>
</tbody>
</table>
Conclusion

- Deep Transform Learning (proposed) outperforms DDL, SDAE and DBN in terms of accuracy.
- Features generated by DTL are good representations because all classifiers KNN, SRC and SVM are able to accurately classify test data.
- Train and Test time is less with proposed technique.
Future Work

- Incorporate stochastic regularization techniques into DTL framework.
- Compare regularized DTL with advanced regularized tools like sparse AutoEncoder, Contractive AutoEncoder, sparse DBN.
- Making supervised DTL framework.
Questions are welcome.