Supervised Deep Learning with Auxiliary Networks

Junbo Zhang†,‡, Guangjian Tian‡, Yadong Mu‡, Wei Fan‡

†School of Information Science and Technology,
Southwest Jiaotong University, Chengdu 610031, China
‡Huawei Noah’s Ark Lab, Hong Kong

August 24–27, 2014, New York, NY, USA
Why Deep Learning?

To model high-level abstractions in data by using architectures composed of multiple non-linear transformations.
Motivations

Deep Learning

Why Deep Learning?
To model high-level abstractions in data by using architectures composed of multiple non-linear transformations.

Very high level representation:
MAN SITTING ...

Slightly higher level representation

Raw input vector representation:
\( \mathbf{x} = [23, 19, 20, 18] \)

Shallow Models

Deep Models

Y. Bengio, Learning Deep Architectures for AI

http://www.wired.com/2013/02/android-neural-network/

Taigman et al., DeepFace: Closing the Gap to Human-Level Performance in Face Verification

Natural language processing:

Speech recognition: Android voice recognition (25% reduction)

Computer vision:

Success in Computer Vision

Experiments

Conclusions
Deep Learning

Why Deep Learning?

To model high-level abstractions in data by using architectures composed of multiple non-linear transformations.

Success in Computer Vision

- Computer vision
  ![Feature maps produced at each layer.](image)
- Speech recognition: Android voice recognition (25% reduction) \(^b\)
- Natural language processing: Machine translation, Matching short text

---

\(^a\) Taigman et al., DeepFace: Closing the Gap to Human-Level Performance in Face Verification

\(^b\) [http://www.wired.com/2013/02/android-neural-network/](http://www.wired.com/2013/02/android-neural-network/)
Existing Deep Learning Schemes

Manners

- Supervised
- Unsupervised
- Semi-supervised
Existing Deep Learning Schemes

Manners
- Supervised
- Unsupervised
- Semi-supervised

Models
- AutoEncoders (AE)
- Restricted Boltzmann Machines (RBM)
- Convolutional Neural Networks
- Recurrent Neural Networks
- ...
Existing Deep Learning Schemes

Manners
- Supervised
- Unsupervised
- Semi-supervised

Models
- AutoEncoders (AE)
- Restricted Boltzmann Machines (RBM)
- Convolutional Neural Networks
- Recurrent Neural Networks
- ...

Deep Architecture (Layer-wise Pre-training)

Stacked Restricted Boltzmann Machines (RBM) → Deep Belief Network (DBN)

Stacked Autoencoders: Unsupervised pre-training + supervised fine-turning
Problems & Shortcoming

1. Sample-specific annotations are always required
2. Ineffectively handle sparse side information

Side information
- More flexible: Similarity/dissimilarity constraints
- Greatly mitigates the workload of annotators
Solution: SUGAR

Main Network is used to reconstruct the input, i.e., the unsupervised autoencoder; Auxiliary Network is used to regularize the learnt network by pairwise similarity or dissimilarity constraints, i.e., the supervised hashing learning; Bridge is used to connect Main Network and Auxiliary Network by enforcing the correlation of their parameters.

\[
\text{Target} \quad \text{X}_l \quad \text{Reconstruction} \\
\text{Labeled data} \quad \text{Auxiliary Network} \quad \text{Training data} \quad \text{Main Network}
\]

- Supervised cost
- Mixed cost
- Unsupervised cost (Reconstruction error)

Bridge
Main Network is used to reconstruct the input, i.e., the unsupervised autoencoder;
Main Network is used to reconstruct the input, \textit{i.e.,} the unsupervised autoencoder;

Auxiliary Network is used to regularize the learnt network by pairwise similarity or dissimilarity constraints, \textit{i.e.,} the supervised hashing learning;
**Solution: SUGAR**

- **Main Network** is used to reconstruct the input, i.e., the unsupervised autoencoder;
- **Auxiliary Network** is used to regularize the learnt network by pairwise similarity or dissimilarity constraints, i.e., the supervised hashing learning;
- **Bridge** is used to connect **Main Network** and **Auxiliary Network** by enforcing the correlation of their parameters.
Main Network

A sparsity-encouraging variant of autoencoder.

Reconstruction: \( \hat{x} \)

Hidden: \( z \)

Input: \( x \)
A sparsity-encouraging variant of autoencoder.

Encoder \( z = f(x) = S_f(Wx + b) \)

Input: \( x \)
Hidden: \( z \)
Reconstruction: \( \hat{x} \)

Objective arg min \( \phi \sum_{x \in X} L(x, \hat{x}) + \lambda \| W \|_{\ell_1} \)

\( \phi = \{W, b, b'\}, W' = W^T \)
A sparsity-encouraging variant of autoencoder.

Encoder  \( z = f(x) = S_f(Wx + b) \)
Decoder  \( \hat{x} = g(z) = S_g(W'z + b') \)
Main Network

A sparsity-encouraging variant of autoencoder.

Encoder \( z = f(x) = S_f(Wx + b) \)

Decoder \( \hat{x} = g(z) = S_g(W'z + b') \)

Reconstruction Error \( \mathcal{L}(x, \hat{x}) = ||x - \hat{x}||^2 \)
Main Network

A sparsity-encouraging variant of autoencoder.

Encoder: $z = f(x) = S_f(Wx + b)$

Decoder: $\hat{x} = g(z) = S_g(W'z + b')$

Reconstruction Error: $\mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2$

Objective: $\arg\min_\phi \sum_{x \in X} \mathcal{L}(x, \hat{x}) + \lambda \|W\|_{\ell_1}$

$\phi = \{W, b, b'\}$, $W' = W^T$. 

\[ \text{Input: } x \]
\[ \text{Hidden: } z \]
\[ \text{Reconstruction: } \hat{x} \]
\[ \text{Encoder: } z = f(x) = S_f(Wx + b) \]
\[ \text{Decoder: } \hat{x} = g(z) = S_g(W'z + b') \]
\[ \text{Reconstruction Error: } \mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2 \]
\[ \text{Objective: } \arg\min_\phi \sum_{x \in X} \mathcal{L}(x, \hat{x}) + \lambda \|W\|_{\ell_1} \]
\[ \phi = \{W, b, b'\}, \quad W' = W^T. \]
Main Network

A sparsity-encouraging variant of autoencoder.

Encoder $z = f(x) = S_f(Wx + b)$
Decoder $\hat{x} = g(z) = S_g(W'z + b')$

Reconstruction Error $\mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2$

Objective $\arg \min_{\phi} \sum_{x \in \mathcal{X}} \mathcal{L}(x, \hat{x}) + \lambda \|W\|_{\ell_1}$

L1 Regularization: Preventing Overfitting
Auxiliary Network

Hashing representation  \( h = H(x) = \text{sgn}(Px + t) \)
Auxiliary Network

Hashing representation \( h = H(x) = \text{sgn}(P x + t) \)

Original objective \( J(P) = \sum_{k=1}^{K} \left\{ \frac{1}{|\mathcal{M}|} \sum_{(x_i, x_j) \in \mathcal{M}} h_k(x_i) h_k(x_j) - \frac{1}{|\mathcal{C}|} \sum_{(x_i, x_j) \in \mathcal{C}} h_k(x_i) h_k(x_j) \right\} \)
Auxiliary Network

Hashing representation \( h = H(x) = \text{sgn}(P x + t) \)

Original objective

\[
J(P) = \sum_{k=1}^{K} \left\{ \frac{1}{|\mathcal{M}|} \sum_{(x_i, x_j) \in \mathcal{M}} h_k(x_i) h_k(x_j) - \frac{1}{|\mathcal{C}|} \sum_{(x_i, x_j) \in \mathcal{C}} h_k(x_i) h_k(x_j) \right\}
\]

Relaxations

\( H(X_l) = \text{sgn}(P X_l) \) is replaced by \( P X_l \)

\[
\Omega_{ij} = \begin{cases} 
1 \times \frac{1}{|\mathcal{M}|}, & (x_i, x_j) \in \mathcal{M}, \\
-1 \times \frac{1}{|\mathcal{C}|}, & (x_i, x_j) \in \mathcal{C}, \\
0, & \text{otherwise}.
\end{cases}
\]
Auxiliary Network

Hashing representation \( h = H(x) = \text{sgn}(Px + t) \)

Original objective \( J(P) = \sum_{k=1}^{K} \left\{ \frac{1}{|M|} \sum_{(x_i, x_j) \in M} h_k(x_i)h_k(x_j) - \frac{1}{|C|} \sum_{(x_i, x_j) \in C} h_k(x_i)h_k(x_j) \right\} \)

Relaxations \( H(X_l) = \text{sgn}(PX_l) \) is replaced by \( PX_l \)

\[ \Omega_{ij} = \begin{cases} 
1 \times \frac{1}{|M|}, & (x_i, x_j) \in M, \\
-1 \times \frac{1}{|C|}, & (x_i, x_j) \in C, \\
0, & \text{otherwise}. 
\end{cases} \]

Relaxed objective

\[ \arg\max_P \frac{1}{2} \text{tr}\{PX_l\Omega X_l^T P^T\}, \]

subject to \( PP^T = I \).

The balancing and pairwise decorrelation constraints can help generate good hash codes in which bits are independent and each bit maximizes the information by generating a balanced partition of the data. They are replaced by the orthogonality constraints.
Bridge: Mixed Objective

\[
\arg \min_{\phi, P} \quad \alpha J_{AE}(\phi) + (1 - \alpha) J_{SH}(P) + \frac{\epsilon}{2} \|P - W\|_F^2 + \lambda \|W\|_{\ell_1}
\]

subject to \(PP^T = I\).

where \(\epsilon\) is a correlation coefficient between \(P\) and \(W\), \(\lambda\) is sparsity (\(L_1\)) penalty ratio, \(\alpha \in [0, 1]\) is a guiding coefficient, and linearly blends the following two objectives:

\[
J_{AE}(\phi) = \sum_{x \in X} \mathcal{L}(x, \hat{x}) = \frac{1}{2} \sum_{x \in X} \|x - \hat{x}\|^2, \quad J_{SH}(P) = -\frac{1}{2} \text{tr}\{PX_l\Omega X_l^T P^T\}.
\]
Bridge: Mixed Objective

\[
\begin{align*}
\arg\min_{\phi, P} & \quad \alpha J_{AE}(\phi) + (1 - \alpha) J_{SH}(P) + \frac{\epsilon}{2} \|P - W\|_F^2 + \lambda \|W\|_{\ell_1} \\
\text{subject to} & \quad PP^T = I.
\end{align*}
\]

where \(\epsilon\) is a correlation coefficient between \(P\) and \(W\), \(\lambda\) is sparsity (\(L_1\)) penalty ratio, \(\alpha \in [0, 1]\) is a guiding coefficient, and linearly blends the following two objectives:

\[
J_{AE}(\phi) = \sum_{x \in X} L(x, \hat{x}) = \frac{1}{2} \sum_{x \in X} \|x - \hat{x}\|^2, \quad J_{SH}(P) = -\frac{1}{2} \text{tr}\{PX\Omega X^T P^T\}.
\]

Alternative Optimization with Stochastic Gradient Descent

- **Fix \(\phi\), Update \(P\)**

  \[
  P \leftarrow P - \eta \frac{\partial J}{\partial P}
  \]

  \[
  P \leftarrow (PP^T)^{-\frac{1}{2}} P \quad \text{(Orthogonal projection)}
  \]

- **Fix \(P\), Update \(\phi\)**

  \[
  \phi \leftarrow \phi - \eta \frac{\partial J}{\partial \phi}
  \]
Extensions: SUGAR with Various Autoencoder

SUGAR with Denoising Autoencoder

\[ \arg \min_{\phi, P} \alpha J_{\text{DAE}}(\phi) + (1 - \alpha) J_{\text{SH}}(P) + \frac{\epsilon}{2} \|P - W\|_F^2 + \lambda \|W\|_{\ell_1}, \]  
subject to \[ PP^T = I. \]

where \[ J_{\text{DAE}}(\phi) = \sum_{\mathbf{x} \in X} \mathbb{E}_{\tilde{x} \sim q(\tilde{x}|x)} \left[ \mathcal{L}(\mathbf{x}, \tilde{x}) \right]. \]

SUGAR with Contractive Autoencoder

\[ \arg \min_{\phi, P} \alpha J_{\text{CAE}}(\phi) + (1 - \alpha) J_{\text{SH}}(P) + \frac{\epsilon}{2} \|P - W\|_F^2 + \lambda \|W\|_{\ell_1}, \]  
subject to \[ PP^T = I. \]

where \[ J_{\text{CAE}}(\phi) = \sum_{\mathbf{x} \in X} \left( \mathcal{L}(\mathbf{x}, \hat{x}) + \mu \|J_f(x)\|_F^2 \right). \]
Deep SUGARs

After training, the feedback decoding modules and the encoder modules with the corresponding classifier modules (all dashed lines) are discarded and the system is used to produce very compact representations by a feed-forward pass through the chain of encoders.
After training, the feedback decoding modules $g$ and the encoder modules $h$ with the corresponding classifier modules (all dashed lines) are discarded and the system is used to produce very compact representations by a feed-forward pass through the chain of encoders $f$. 

Layer-wise Training
Experiments: Datasets

  - Variations on MNIST
  - Discrimination between tall and wide rectangles
  - Recognition of convex sets

Table 1: Datasets

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Train</th>
<th>Valid.</th>
<th>Test</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>50000</td>
<td>10000</td>
<td>10000</td>
<td>10</td>
</tr>
<tr>
<td>Rectangles</td>
<td>1000</td>
<td>200</td>
<td>50000</td>
<td>2</td>
</tr>
<tr>
<td>Rect_Img</td>
<td>10000</td>
<td>2000</td>
<td>50000</td>
<td>2</td>
</tr>
<tr>
<td>Convex</td>
<td>7000</td>
<td>1000</td>
<td>50000</td>
<td>2</td>
</tr>
<tr>
<td>MNIST_Basic</td>
<td>10000</td>
<td>2000</td>
<td>50000</td>
<td>10</td>
</tr>
<tr>
<td>MNIST_Rot</td>
<td>10000</td>
<td>2000</td>
<td>50000</td>
<td>10</td>
</tr>
<tr>
<td>MNIST_Rand</td>
<td>10000</td>
<td>2000</td>
<td>50000</td>
<td>10</td>
</tr>
<tr>
<td>MNIST_Img</td>
<td>10000</td>
<td>2000</td>
<td>50000</td>
<td>10</td>
</tr>
<tr>
<td>MNIST_RotImg</td>
<td>10000</td>
<td>2000</td>
<td>50000</td>
<td>10</td>
</tr>
</tbody>
</table>
Baseline methods

- **SVM**
  - SVM-RBF: SVM with RBF kernels
  - SVM-Poly: SVM with polynomial kernels
- **NNet**: Feed-forward neural network
- **GSM**: Gated softmax classifier
- **NonGSM**: Non-factored gated softmax classifier
- **SAA**: Stacked Autoassociator Network
- **RBM**: Restricted Boltzmann Machine
Performance Evaluation: Shallow Architecture on MNIST

- **Motivations**
- **Experiments**
- **Conclusions**
Performance Evaluation: Shallow Architecture on MNIST

Effect of Different Factors

(a) Guiding Coefficient

(b) Labeled Data

(c) Sparsity Penalty
Performance Evaluation: Shallow Architecture on MNIST

Effect of Different Factors

(a) Guiding Coefficient
(b) Labeled Data
(c) Sparsity Penalty

Filters learnt from MNIST with various sparsity

(d) 10%
(e) 25%
(f) 40%
(g) 50%
(h) 60%
(i) 70%
Performance Evaluation: Shallow Architecture

Guidance to Autoencoder Variants (DAE and CAE)

(a) DAE vs. SUGAR
(b) CAE vs. SUGAR

Figure 2: Guiding ability on autoencoder variants
Deep Architecture on Benchmark Classification Tasks

Figure 3: Classification error rates

<table>
<thead>
<tr>
<th>Dataset/Model:</th>
<th>SVM-RBF</th>
<th>SVM-Poly</th>
<th>NNet</th>
<th>GSM</th>
<th>NonGSM</th>
<th>SAA-3</th>
<th>RBM</th>
<th>SUGAR-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rectangles</td>
<td>02.15</td>
<td>02.15</td>
<td>07.16</td>
<td>0.83</td>
<td><strong>0.56</strong></td>
<td>02.41</td>
<td>04.71</td>
<td>03.49</td>
</tr>
<tr>
<td>Rect_{Img}</td>
<td>24.04</td>
<td>24.05</td>
<td>33.20</td>
<td><strong>22.51</strong></td>
<td>23.17</td>
<td>24.05</td>
<td>23.69</td>
<td><strong>22.55</strong></td>
</tr>
<tr>
<td>Convex</td>
<td>19.13</td>
<td>19.82</td>
<td>32.25</td>
<td>17.08</td>
<td>21.03</td>
<td>18.41</td>
<td>19.92</td>
<td><strong>17.00</strong></td>
</tr>
<tr>
<td>MNIST_{Basic}</td>
<td><strong>03.03</strong></td>
<td>03.69</td>
<td>04.69</td>
<td>03.70</td>
<td>03.98</td>
<td>03.46</td>
<td>03.94</td>
<td>03.47</td>
</tr>
<tr>
<td>MNIST_{Rot}</td>
<td>11.11</td>
<td>15.42</td>
<td>18.11</td>
<td>11.75</td>
<td>16.15</td>
<td>10.30</td>
<td>14.69</td>
<td><strong>9.53</strong></td>
</tr>
<tr>
<td>MNIST_{Rand}</td>
<td>14.58</td>
<td>16.62</td>
<td>20.04</td>
<td>10.48</td>
<td>11.89</td>
<td>11.28</td>
<td><strong>09.80</strong></td>
<td>11.40</td>
</tr>
<tr>
<td>MNIST_{Img}</td>
<td>22.61</td>
<td>24.01</td>
<td>27.41</td>
<td>23.65</td>
<td>22.07</td>
<td>23.00</td>
<td><strong>16.15</strong></td>
<td>20.65</td>
</tr>
<tr>
<td>MNIST_{RotImg}</td>
<td>55.18</td>
<td>56.41</td>
<td>62.16</td>
<td>55.82</td>
<td>55.16</td>
<td>51.93</td>
<td>52.21</td>
<td><strong>49.40</strong></td>
</tr>
<tr>
<td>Average</td>
<td>18.98</td>
<td>20.27</td>
<td>25.63</td>
<td>18.23</td>
<td>19.25</td>
<td>18.11</td>
<td>18.14</td>
<td><strong>17.19</strong></td>
</tr>
</tbody>
</table>
Conclusions

Proposed model: SUGAR

- SUGAR incorporates both weak supervision (pairwise constraints) or strong supervision (labeled) into Autoencoder framework
- It is demonstrated that both semi-supervised and supervised SUGAR is consistently more accurate than unsupervised autoencoder

Potential Application Areas

1. Handwriting Recognition
2. Domain Adaptation
3. Telecommunication Data Mining
4. Others
   - Multi-source data
   - Few Labeled data

Q & A

Thanks