Marginalized Stacked Denoising Autoencoders for Domain Adaptation

**Minmin Chen**, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, Fei Sha
Domain Adaptation

**Training:**
Source Domain
Sufficient data

**Testing:**
Target Domain
Insufficient data

\[ P_S(Y|X) \]
\[ P_h(Y|X) \]
\[ P_T(Y|X) \]
Showcase application: Sentiment Analysis

predict sentiment (star rating) from review text

[Blitzer et al., 2006]
Same Task, Different Distribution

Naïve approach: train on source, test on target

Source domain: Books
(Test Error: 13%)

Target domain: Kitchen Appliances
(Test Error: 24%)

Minmin Chen, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, Fei Sha
Marginalized Stacked Denoising Autoencoders
I read 2-3 books a week, and this is without a doubt my favorite of this year. A beautiful novel by Afghan-American Khaled Hosseini that ranks among the best-written and provocative stories of the year so far. This unusually eloquent story is also about the fragile relationship ....

This unit makes the best coffee I've had in a home. It is my favorite unit. It makes excellent and HOT coffee. The carafe is solidly constructed and fits securely in square body. The timer is easy to program, as is the clock.
Learning Joint Feature Representation

[Blitzer et al., 2007; Satpal and Sarawagi, 2007; Pan et al., 2010]
Denoising Autoencoders (DA)

[Vincent et al., 2008; Glorot et al., 2011]

I read 2-3 books a week, and this is without a doubt my favorite of this year. A beautiful novel by Afghan-American Khaled Hosseini that ranks among the best-written and provocative of the year so far. This unusually eloquent story is also about the relationship....

Minmin Chen, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, Fei Sha
Denoising Autoencoder (DA)

[Vincent et al., 2008]

\[ \text{min}_W \| x - g(W^T f(W\tilde{x})) \|^2 \]

Minmin Chen, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, Fei Sha
Denoising Autoencoder (DA)

[Vincent et al., 2008]

Minmin Chen, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, Fei Sha
SDA for Feature Learning

\[ \text{x} \rightarrow h_1 \rightarrow h_2 \rightarrow \text{new representation} \]

[Glorot et al., 2011]

linear classifier (SVM)
Accurate but Slow

- SDAs generate robust features for domain adaptation
- Pre-training of SDAs requires (stochastic) gradient descent, slow for large scale dataset
  - dense-matrix GPU implementation [Bergstra, et al., 2010]
  - reconstruction sampling [Dauphin et al., 2011]
- hyper-parameters (learning rate, number of epochs, noise ratio, mini-batch size, network structure, etc.) [Bergstra and Bengio, 2012]
Research Goal

![Graph showing training time vs. transfer ratio]

- **Bag-of-words** (0 secs)
- **PCA** (~3 mins)
- **SCL** (47 secs)
- **CODA** (~25 mins)
- **SDA** (~5 hours)

Keep the **accuracy**
Improve the **speed**

Minmin Chen, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, Fei Sha
marginalized Denoising Autoencoder (mDA)

Convexity

\[ x \xrightarrow{f_W} z \xleftarrow{g_W} \tilde{x} \]

\[\ell_{DA} = \sum_{i=1}^{n} \| x_i - g(W^T f(W\tilde{x})) \|^2\]

\[\ell = \sum_{i=1}^{n} \| x_i - W\tilde{x}_i \|^2\]

local minima

global minima
marginalized Denoising Autoencoder (mDA)
Closed-form solution

\[ \tilde{\mathbf{x}} = \mathbf{W} \mathbf{z} \]

Minimization:
\[ \ell = \sum_{i=1}^{n} \| \mathbf{x}_i - \mathbf{W} \tilde{\mathbf{x}}_i \|^2 \]

Solution:
\[ \mathbf{W} = \left( \sum_{i=1}^{n} \mathbf{x}_i \tilde{\mathbf{x}}_i^\top \right) \left( \sum_{i=1}^{n} \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top \right)^{-1} \]
marginalized Denoising Autoencoder (mDA)

Multiple corruptions

\[
W = \left( \sum_{i=1}^{n} x_i \tilde{x}_i^\top \right) \left( \sum_{i=1}^{n} \tilde{x}_i \tilde{x}_i^\top \right)^{-1}
\]

create \( m \) corruptions for each input

\[
\ell = \sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} \| x_i - W \tilde{x}_{i,j} \|^2
\]

\[
W = \left( \sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} x_i \tilde{x}_{i,j}^\top \right) \left( \sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} \tilde{x}_{i,j} \tilde{x}_{i,j}^\top \right)^{-1}
\]
marginalized Denoising Autoencoder (mDA)

Marginalized corruption

\[
\begin{align*}
\tilde{x} &\rightarrow W \rightarrow z \\
\sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} x_i \tilde{x}_{i,j} &\approx \frac{1}{m} \sum_{i=1}^{n} \sum_{j=1}^{m} \tilde{x}_{i,j} \tilde{x}_{i,j}^\top \\
W &\approx \left( \sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} x_i \tilde{x}_{i,j} \right)^{-1} \left( \sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} \tilde{x}_{i,j} \tilde{x}_{i,j}^\top \right)^{-1}
\end{align*}
\]
marginalized Denoising Autoencoder (mDA)
Marginalized corruption

\[
\tilde{x} \quad W \quad z
\]

\[
W = \left( \sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} x_i \tilde{x}_{i,j}^T \right) \left( \sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} \tilde{x}_{i,j} \tilde{x}_{i,j}^T \right)^{-1}
\]

\[
W^* = \left( \sum_{i=1}^{n} E[x_i \tilde{x}_i^T] \right) \left( \sum_{i=1}^{n} E[\tilde{x}_i \tilde{x}_i^T] \right)^{-1}
\]

- closed form!
- corruption marginalized out!

\[
P_{\alpha,\beta} = S_{\alpha \beta} q_{\beta}
\]

\[
Q_{\alpha,\beta} = S_{\alpha \beta} q_{\alpha} q_{\beta}
\]
marginalized Stacked Denoising Autoencoder (mSDA)
mSDA for Feature Learning

new representation

linear classifier (SVM)
20 lines of Matlab

Algorithm 1 mDA in MATLAB™.

```matlab
function [W,h]=mDA(X,p);
    X=[X;ones(1,size(X,2))];
    d=size(X,1);
    q=ones(d,1).*(1-p); q(end)=1;
    S=X*X';
    Q=S.*(q*q');
    Q(1:d+1:end)=q.*diag(S);
    P=S.*repmat(q,1,d);
    W=((Q+1e-5*eye(d))\P(:,:,1:end-1))';
    h=W*X;
end
```

Algorithm 2 mSDA in MATLAB™.

```matlab
function [Ws,hs]=mSDA(X,p,l);
    [d,n]=size(X);
    Ws=zeros(d,d+1,l);
    hs=zeros(d,d+l+1);
    hs(:,:,1)=X;
    for t=1:l
        Ws(:,:,t)=mDA(hs(:,:,t),p);
        hst=Ws(:,:,t)*[hs(:,:,t);ones(1,n)];
        hs(:,:,t+1)=tanh(hst);
    end;
end
```
Results: Amazon review dataset

[Blitzer et al., 2006]
Experimental Results

\[
\text{transfer loss} = \text{error}(B \rightarrow E) - \text{error}(D \rightarrow E)
\]
Transfer ratio vs. training time

4 domains, 12 tasks
\(d = 5,000\)
\(n = 27,677\)

Minmin Chen, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, Fei Sha
Marginalized Stacked Denoising Autoencoders
Large data set

20 domains, 380 tasks
\( d = 5,000 \)
\( n = 339,675 \)

---

**Marginalized Stacked Denoising Autoencoders**
High Dimensional Data

4 domains, 12 tasks
\( d = 5,000 - 40,000 \)
\( n = 27,677 \)

Minmin Chen, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, Fei Sha

Marginalized Stacked Denoising Autoencoders

Thursday, April 19, 12
Conclusion

- **m**arginalized **S**tacked **D**enoising **A**utoencoder (**mSDA**)  
  - marginalizes out corruption  
  - keeps high accuracy of SDAs but is orders of magnitudes faster  

**Optimization:**  
- is layer-wise convex  
- has layer-wise closed form solutions  
- is easy to implement
Conclusion

- **marginalized** Stacked Denoising Autoencoder (**mSDA**)
  - marginalizes out corruption
  - keeps high accuracy of SDAs but is orders of magnitudes faster

- Optimization:
  - is layer-wise convex
  - has layer-wise closed form solutions
  - is easy to implement
Conclusion

- **marginalized Stacked Denoising Autoencoder (mSDA)**
  - marginalizes out corruption
  - keeps high accuracy of SDAs but is orders of magnitudes faster

Optimization:
- is layer-wise convex
- has layer-wise closed form solutions
- is easy to implement
Conclusion

- **marginalized Stacked Denoising Autoencoder (mSDA)**
  - marginalizes out corruption
  - keeps high accuracy of SDAs but is orders of magnitudes faster
- Optimization:
  - *is layer-wise convex*
  - has layer-wise closed form solutions
  - is easy to implement
Conclusion

- **marginalized Stacked Denoising Autoencoder (mSDA)**
  - marginalizes out corruption
  - keeps high accuracy of SDAs but is orders of magnitudes faster

Optimization:

- is layer-wise convex
- **has layer-wise closed form solutions**
- is easy to implement
Conclusion

- **marginalized Stacked Denoising Autoencoder (mSDA)**
  - marginalizes out corruption
  - keeps high accuracy of SDAs but is orders of magnitudes faster

- **Optimization:**
  - is layer-wise convex
  - has layer-wise closed form solutions
  - is easy to implement

```
Algorithm 1 mDA in MATLAB\textsuperscript{TM}.
function [W,h]=mDA(X,p);
X=[X;ones(1,size(X,2))];
d=size(X,1);
g=ones(d,1).*(1-p); q(end)=1;
S=X*X';
Q=S.*q';
Q(1:d+1:end)=q.*diag(S);
P=S.*repmat(q,1,d);
W=(Q+1e-5*eye(d))\P(:,1:end-1)';
h=W*X;
```

```
Algorithm 2 mSDA in MATLAB\textsuperscript{TM}.
function [h,hs]=mSDA(X);
[d,n]=size(X);
Ws=zeros(d,d+1,1);
hs=zeros(d,d,l+1);
hs(:,:,1)=X;
for t=1:l
  Ws(:,:,t)=mDA(hs(:,:,t),p);
  hst=Ws(:,:,t)*[hs(:,:,t);ones(1,n)];
  hs(:,:,t+1)=tanh(hst);
end;
```
Thank you!

Questions?