Stochastic Neighbor Compression

Matt J. Kusner
Stephen Tyree
Kilian Q. Weinberger
Kunal Agrawal

Washington University in St. Louis
NN Classifier

[Cover & Hart, 1967]
NN Classifier

[Cover & Hart, 1967]

- test input

- class 1

- class 2

- class 3
NN Classifier

1-nearest neighbor rule

[Cover & Hart, 1967]
NN Classifier

1-nearest neighbor rule

[c]over & Hart, 1967

complexity:

\[ O(dn) \]

\[ n \] training inputs

\[ d \] features
**NN Classifier**

l-nearest neighbor rule

**complexity:**

\[ O(dn) \]

\( n \) training inputs

\( d \) features

for each test input!

\[ n = 6 \times 10^4 \]

\[ d = 784 \]

\[ O(dn) \]

\( \approx 47 \text{ million} \)

[Cover & Hart, 1967]
NN Classifier

1-nearest neighbor rule

complexity:
$O(dn)$

$n$ training inputs
$d$ features

How can we reduce this complexity?
NN Classifier

1-nearest neighbor rule

complexity: $O(dn)$
$n$: training inputs
$d$: features

How can we reduce this complexity?
1. reduce $n$
NN Classifier

1-nearest neighbor rule

complexity:

\[ O(dn) \]

\( n \) training inputs
\( d \) features

How can we reduce this complexity?

1. reduce \( n \)

Training Consistent Sampling
[Hart, 1968]
[Anguilli, 2005]

Prototype Generation
[Bandyopadhyay & Maulik, 2002]
[Mollineda et al., 2002]

Prototype Positioning
[Bermejo & Cabestany, 1999]
[Toussaint 2002]
NN Classifier

1-nearest neighbor rule

complexity:
\[ O(dn) \]

- \( n \) training inputs
- \( d \) features

How can we reduce this complexity?

1. reduce \( n \)
2. reduce \( d \)
NN Classifier

1-nearest neighbor rule

<table>
<thead>
<tr>
<th>complexity:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(dn)$</td>
</tr>
</tbody>
</table>

$n$ training inputs
$d$ features

How can we reduce this complexity?

1. reduce $n$
2. reduce $d$

Dimensionality Reduction

[Tenenbaum et al., 2000]
[Hinton & Roweis, 2002]
[Weinberger et al., 2004]
[van der Maaten & Hinton, 2008]
[Weinberger & Saul, 2009]
NN Classifier

1-nearest neighbor rule

How can we reduce this complexity?
1. reduce $n$
2. reduce $d$
3. use data structures

complexity:

$O(dn)$

$n$ training inputs

$d$ features
NN Classifier

1-nearest neighbor rule

complexity:
\[ O(dn) \]

\( n \) training inputs
\( d \) features

How can we reduce this complexity?

1. reduce \( n \)
2. reduce \( d \)
3. use data structures

Tree Structures
[Omohundro, 1989]
[Beygelzimer et al., 2006]

Hashing
[Gionis et al., 1999]
[Andoni & Indyk, 2006]
NN Classifier

1-nearest neighbor rule

complexity:
\[ O(dn) \]

- \( n \) training inputs
- \( d \) features

How can we reduce this complexity?

1. reduce \( n \)
2. reduce \( d \)
3. use data structures
NN Classifier

1-nearest neighbor rule

**complexity:**

\[ O(dn) \]

- \( n \) training inputs
- \( d \) features

How can we reduce this complexity?

1. reduce \( n \)
2. reduce \( d \)
3. use data structures

Dataset Compression
Main Idea
learn new synthetic inputs!

training data

‘compressed’ data
Main Idea
learn new synthetic inputs!

training data

\[
\begin{bmatrix}
X_1 & X_2 & \cdots & X_i & \cdots & X_{n-1} & X_n \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
Y_1 & Y_2 & \cdots & Y_i & \cdots & Y_{n-1} & Y_n \\
\end{bmatrix}
\]

‘compressed’ data

\[
\begin{bmatrix}
Z_1 & \cdots & Z_m \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\hat{Y}_1 & \cdots & \hat{Y}_m \\
\end{bmatrix}
\]
Main Idea
learn new synthetic inputs!

$m << n$

training data

\begin{bmatrix}
X_1 & X_2 & \cdots & X_i & \cdots & X_{n-1} & X_n \\
Y_1 & Y_2 & \cdots & Y_i & \cdots & Y_{n-1} & Y_n
\end{bmatrix}

‘compressed’ data

\begin{bmatrix}
Z_1 & \cdots & Z_m \\
\hat{Y}_1 & \cdots & \hat{Y}_m
\end{bmatrix}
Main Idea

learn new synthetic inputs!

use ‘compressed’ set to make future predictions

‘compressed’ data

\[
\begin{bmatrix}
z_1 & \cdots & z_m \\
\hat{y}_1 & \cdots & \hat{y}_m
\end{bmatrix}
\]
Approach
steps to a new training set:
Approach

steps to a new training set:
1. subsample
Approach

steps to a new training set:
1. subsample
2. learn prototypes
Approach

steps to a new training set:
1. subsample
2. learn prototypes

move inputs to minimize 1-nn training error
Learning Prototypes

move inputs to minimize 1-nn training error
Learning Prototypes

move inputs to minimize 1-nn training error

\[
\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} [\text{nn}_{\{z_1, \ldots, z_m\}}(x_i) \neq y_i]
\]

label of nearest neighbor to \( x_i \) in set \( \{Z_1, \ldots, Z_m\} \)

true label
Learning Prototypes

move inputs to minimize 1-nn training error

$$\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} \left[ \text{nn}\{z_1, \ldots, z_m\}(x_i) \neq y_i \right]$$

**label of nearest neighbor to** $x_i$ **in set** \(\{Z_1, \ldots, Z_m\}\)

**true label**
Learning Prototypes

move inputs to minimize 1-nn training error

$$\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} [\min_{j=1}^{n} \{z_j \neq y_i\}]$$

not continuous
not differentiable
Learning Prototypes

move inputs to minimize 1-nn training error

$$\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} \left[ \min_{j=1}^{k} \{ \langle x_i \rangle \} \neq y_i \right]$$

Stochastic Neighborhood
[Hinton & Roweis, 2002]
Learning Prototypes

move inputs to minimize $1$-nn training error

$$\min_{z_1,\ldots,z_m} \sum_{i=1}^{n} \left[ \min_{j \neq y_i} \{(x_i) \neq y_j\} \right]$$

Stochastic Neighborhood
[Hinton & Roweis, 2002]
Learning Prototypes

move inputs to minimize 1-nn training error

\[
\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} \left[ \min \{1, \ldots, m\} (x_i) \neq y_i \right]
\]

probability \( x_i \) is predicted correctly

Stochastic Neighborhood
[Hinton & Roweis, 2002]
Learning Prototypes

move inputs to minimize 1-nn training error

\[
\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} \left[ \min_{j} \{ z_j \} (x_i) \neq y_i \right]
\]

probability \( x_i \) is predicted correctly

\[
\Delta \triangleq \frac{1}{Z} \sum_{j: \hat{y}_j = y_i} \exp(-\gamma^2 \| x_i - z_j \|_2^2)
\]

[Hinton & Roweis, 2002]
Learning Prototypes

move inputs to minimize 1-nn training error

$$\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} \left[ \min_{j} \{1, \ldots, m\} \{x_i \neq y_i\} \right]$$

minimize negative log likelihood

$$\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} - \log(p_i)$$

probability $x_i$ is predicted correctly

$$\triangleq \frac{1}{Z} \sum_{j: \hat{y}_j = y_i} \exp(-\gamma^2 \|x_i - z_j\|_2^2)$$
Learning Prototypes

move inputs to minimize 1-nn training error

\[
\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} \left[ \min \{ y_j : y_j \neq y_i \} (x_i) \right]
\]

use conjugate gradient descent!

minimize negative log likelihood

\[
\min_{z_1, \ldots, z_m} \sum_{i=1}^{n} - \log(p_i)
\]

probability \( x_i \) is predicted correctly

\[
\triangleq \frac{1}{Z} \sum_{j: \hat{y}_j = y_i} \exp(-\gamma^2 \| x_i - z_j \|^2_2)
\]
Results
[a motivating visualization]
Results

Yale-Faces

- 38 people
- ~64 images/person
- lighting changes
Results

- 38 people
- ~64 images/person
- lighting changes
Results

Yale-Faces

- 38 people
- ~64 images/person
- lighting changes
Results

- 38 people
- ~64 images/person
- lighting changes

Yale-Faces
Results

subsampling real faces
Results

subsampling real faces
Results
Results

learned faces
Results
[datasets]
Results

Table 1. Characteristics of datasets used in evaluation.

<table>
<thead>
<tr>
<th>DATASET STATISTICS</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>YALE-FACES</td>
</tr>
<tr>
<td>ISOLET</td>
</tr>
<tr>
<td>LETTERS</td>
</tr>
<tr>
<td>ADULT</td>
</tr>
<tr>
<td>W8A</td>
</tr>
<tr>
<td>MNIST</td>
</tr>
<tr>
<td>FOREST</td>
</tr>
</tbody>
</table>
Results

training inputs

Table 1. Characteristics of datasets used in evaluation.

| Dataset          | Statistics | \( n \) | \( |\mathcal{Y}| \) | \( d \) (\( d_L \)) |
|------------------|------------|--------|----------------|------------------|
| YALE-FACES       |            | 1961   | 38             | 8064 (100)       |
| ISOLET           |            | 3898   | 26             | 617 (172)        |
| LETTERS          |            | 16000  | 26             | 16 (16)          |
| ADULT            |            | 32562  | 2              | 123 (50)         |
| W8A              |            | 49749  | 2              | 300 (100)        |
| MNIST            |            | 60000  | 10             | 784 (164)        |
| FOREST           |            | 100000 | 7              | 54 (54)          |
Table 1. Characteristics of datasets used in evaluation.

<table>
<thead>
<tr>
<th>DATASET STATISTICS</th>
<th>NAME</th>
<th>$n$</th>
<th>$\mathcal{Y}$</th>
<th>$d$ ($d_L$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>YALE-FACES</td>
<td>1961</td>
<td>38</td>
<td>8064 (100)</td>
</tr>
<tr>
<td></td>
<td>ISOLET</td>
<td>3898</td>
<td>26</td>
<td>617 (172)</td>
</tr>
<tr>
<td></td>
<td>LETTERS</td>
<td>16000</td>
<td>26</td>
<td>16 (16)</td>
</tr>
<tr>
<td></td>
<td>ADULT</td>
<td>32562</td>
<td>2</td>
<td>123 (50)</td>
</tr>
<tr>
<td></td>
<td>W8A</td>
<td>49749</td>
<td>2</td>
<td>300 (100)</td>
</tr>
<tr>
<td></td>
<td>MNIST</td>
<td>60000</td>
<td>10</td>
<td>784 (164)</td>
</tr>
<tr>
<td></td>
<td>FOREST</td>
<td>100000</td>
<td>7</td>
<td>54 (54)</td>
</tr>
</tbody>
</table>
Results

Table 1. Characteristics of datasets used in evaluation.

| NAME          | n    | |Y|   | d (d_L)   |
|---------------|------|------|------|-----------|
| YALE-FACES    | 1961 | 38   | 8064 (100) |
| ISOLET        | 3898 | 26   | 617 (172)  |
| LETTERS       | 16000| 26   | 16 (16)    |
| ADULT         | 32562| 2    | 123 (50)   |
| W8A           | 49749| 2    | 300 (100)  |
| MNIST         | 60000| 10   | 784 (164)  |
| FOREST        | 100000| 7    | 54 (54)    |
Results

original & reduced features [Weinberger & Saul, 2009]

Table 1. Characteristics of datasets used in evaluation.

<table>
<thead>
<tr>
<th>DATASET</th>
<th>n</th>
<th></th>
<th></th>
<th>d (d_L)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YALE-FACES</td>
<td>1961</td>
<td>38</td>
<td></td>
<td>8064 (100)</td>
</tr>
<tr>
<td>ISOLET</td>
<td>3898</td>
<td>26</td>
<td></td>
<td>617 (172)</td>
</tr>
<tr>
<td>LETTERS</td>
<td>16000</td>
<td>26</td>
<td></td>
<td>16 (16)</td>
</tr>
<tr>
<td>ADULT</td>
<td>32562</td>
<td>2</td>
<td></td>
<td>123 (50)</td>
</tr>
<tr>
<td>W8A</td>
<td>49749</td>
<td>2</td>
<td></td>
<td>300 (100)</td>
</tr>
<tr>
<td>MNIST</td>
<td>60000</td>
<td>10</td>
<td></td>
<td>784 (164)</td>
</tr>
<tr>
<td>FOREST</td>
<td>100000</td>
<td>7</td>
<td></td>
<td>54 (54)</td>
</tr>
</tbody>
</table>
Results

[error]
Results
Results

ratio of training inputs in compressed set
Results

Test error
Results
Results on full training set
Results

![Graph showing the relationship between compression ratio and error for different methods: KNN without LMNN, KNN with LMNN, subsampling, CNN (Hart, 1968), FCNN (Angiulli, 2005), and SNC. The graph is labeled 'isolet' and shows how error decreases as compression ratio increases for each method.]
Results

Training set consistent sampling
Results

![Graph showing error vs compression ratio for different methods including KNN without LMNN, KNN (with LMNN), subsampling, CNN (Hart, 1968), FCNN (Angiulli, 2005), and our method.]
Results
[test-time speed-up]
Results
[test-time speed-up]

complementary approaches
ball-trees
locality-sensitive hashing (LSH)
Results
[test-time speed-up]

complementary approaches
ball-trees
locality-sensitive hashing (LSH)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>1%</th>
</tr>
</thead>
<tbody>
<tr>
<td>YALE-FACES</td>
<td>--</td>
</tr>
<tr>
<td>ISOLET</td>
<td>76</td>
</tr>
<tr>
<td>LETTERS</td>
<td>143</td>
</tr>
<tr>
<td>ADULT</td>
<td>156</td>
</tr>
<tr>
<td>W8A</td>
<td>146</td>
</tr>
<tr>
<td>MNIST</td>
<td>136</td>
</tr>
<tr>
<td>FOREST</td>
<td>146</td>
</tr>
</tbody>
</table>
Results
[test-time speed-up]

complementary approaches
ball-trees
locality-sensitive hashing (LSH)

<table>
<thead>
<tr>
<th>DATASET</th>
<th>1-NN full dataset</th>
<th>1-NN after SNC</th>
</tr>
</thead>
<tbody>
<tr>
<td>YALE-FACES</td>
<td>76</td>
<td>23</td>
</tr>
<tr>
<td>ISOLET</td>
<td>143</td>
<td>9.3</td>
</tr>
<tr>
<td>LETTERS</td>
<td>156</td>
<td>56</td>
</tr>
<tr>
<td>ADULT</td>
<td>146</td>
<td>68</td>
</tr>
<tr>
<td>W8A</td>
<td>136</td>
<td>54</td>
</tr>
<tr>
<td>MNIST</td>
<td>146</td>
<td>3.1</td>
</tr>
<tr>
<td>FOREST</td>
<td>146</td>
<td>12</td>
</tr>
</tbody>
</table>

compression rate
Results
[test-time speed-up]

complementary approaches
ball-trees
locality-sensitive hashing (LSH)

<table>
<thead>
<tr>
<th>DATASET</th>
<th>1%</th>
</tr>
</thead>
<tbody>
<tr>
<td>YALE-FACES</td>
<td>--</td>
</tr>
<tr>
<td>ISOLET</td>
<td>76</td>
</tr>
<tr>
<td>LETTERS</td>
<td>143</td>
</tr>
<tr>
<td>ADULT</td>
<td>156</td>
</tr>
<tr>
<td>W8A</td>
<td>146</td>
</tr>
<tr>
<td>MNIST</td>
<td>136</td>
</tr>
<tr>
<td>FOREST</td>
<td>146</td>
</tr>
</tbody>
</table>

ball-trees full dataset
ball-trees after SNC

compression rate
Results
[test-time speed-up]

complementary approaches
ball-trees
locality-sensitive hashing (LSH)

<table>
<thead>
<tr>
<th>DATASET</th>
<th>LSH full dataset</th>
<th>LSH after SNC</th>
</tr>
</thead>
<tbody>
<tr>
<td>YALE-FACES</td>
<td>76</td>
<td>13</td>
</tr>
<tr>
<td>ISOLET</td>
<td>143</td>
<td>39</td>
</tr>
<tr>
<td>LETTERS</td>
<td>156</td>
<td>100</td>
</tr>
<tr>
<td>ADULT</td>
<td>146</td>
<td>84</td>
</tr>
<tr>
<td>W8A</td>
<td>136</td>
<td>39</td>
</tr>
<tr>
<td>MNIST</td>
<td>146</td>
<td>3.1</td>
</tr>
<tr>
<td>FOREST</td>
<td>136</td>
<td>12</td>
</tr>
</tbody>
</table>
Results

[test-time speed-up]

complementary approaches

ball-trees

locality-sensitive hashing (LSH)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>1%</th>
<th>2%</th>
<th>4%</th>
<th>8%</th>
<th>16%</th>
</tr>
</thead>
<tbody>
<tr>
<td>YALE-FACES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISOLET</td>
<td>76</td>
<td>47</td>
<td>26</td>
<td>14</td>
<td>7.0</td>
</tr>
<tr>
<td>LETTERS</td>
<td>143</td>
<td>73</td>
<td>34</td>
<td>16</td>
<td>7.6</td>
</tr>
<tr>
<td>ADULT</td>
<td>156</td>
<td>75</td>
<td>36</td>
<td>17</td>
<td>7.8</td>
</tr>
<tr>
<td>W8A</td>
<td>146</td>
<td>71</td>
<td>33</td>
<td>15</td>
<td>7.3</td>
</tr>
<tr>
<td>MNIST</td>
<td>136</td>
<td>66</td>
<td>32</td>
<td>15</td>
<td>7.1</td>
</tr>
<tr>
<td>FOREST</td>
<td>146</td>
<td>70</td>
<td>32</td>
<td>15</td>
<td>—</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>1%</th>
<th>2%</th>
<th>4%</th>
<th>8%</th>
<th>16%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compression Ratio</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YALE-FACES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISOLET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LETTERS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADULT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W8A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MNIST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FOREST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SPEED-UP
Results
[test-time speed-up]

complementary approaches
ball-trees
locality-sensitive hashing (LSH)

<table>
<thead>
<tr>
<th>Datasets</th>
<th>1%</th>
<th>2%</th>
<th>4%</th>
<th>8%</th>
<th>16%</th>
</tr>
</thead>
<tbody>
<tr>
<td>YALE-FACES</td>
<td>-</td>
<td>28</td>
<td>19</td>
<td>12</td>
<td>6.5</td>
</tr>
<tr>
<td>ISOLET</td>
<td>76</td>
<td>17</td>
<td>11</td>
<td>7.3</td>
<td>7.0</td>
</tr>
<tr>
<td>LETTERS</td>
<td>143</td>
<td>13</td>
<td>6.8</td>
<td>3.7</td>
<td>7.6</td>
</tr>
<tr>
<td>ADULT</td>
<td>156</td>
<td>28</td>
<td>15</td>
<td>7.3</td>
<td>7.8</td>
</tr>
<tr>
<td>W8A</td>
<td>146</td>
<td>36</td>
<td>26</td>
<td>15</td>
<td>7.3</td>
</tr>
<tr>
<td>MNIST</td>
<td>136</td>
<td>29</td>
<td>16</td>
<td>8.4</td>
<td>7.1</td>
</tr>
<tr>
<td>FOREST</td>
<td>146</td>
<td>1.6</td>
<td>0.90</td>
<td>1.1</td>
<td>-</td>
</tr>
</tbody>
</table>
Summary

Stochastic Neighbor Compression
- learns a compressed training set for NN classifier
Summary

Stochastic Neighbor Compression

- learns a compressed training set for NN classifier

Stochastic Neighborhood

[Hinton & Roweis, 2002]
Stochastic Neighbor Compression

- learns a compressed training set for NN classifier

- Stochastic Neighborhood
  [Hinton & Roweis, 2002]

- Compression by 96% without error increase in 5/7 cases
Summary

Stochastic Neighbor Compression
- learns a compressed training set for NN classifier
- [Stochastic Neighborhood](#) [Hinton & Roweis, 2002]
- Compression by 96% without error increase in 5/7 cases
- test-time speed-ups on top of NN data structures
Thank you! Questions?
Results
[robustness to label noise]
Results
[robustness to label noise]
Results

[robustness to label noise]

noise added to training labels
Results

[robustness to label noise]

using larger $k$
Results

[robustness to label noise]

testing set
consistent sampling
Results
[robustness to label noise]

Our method