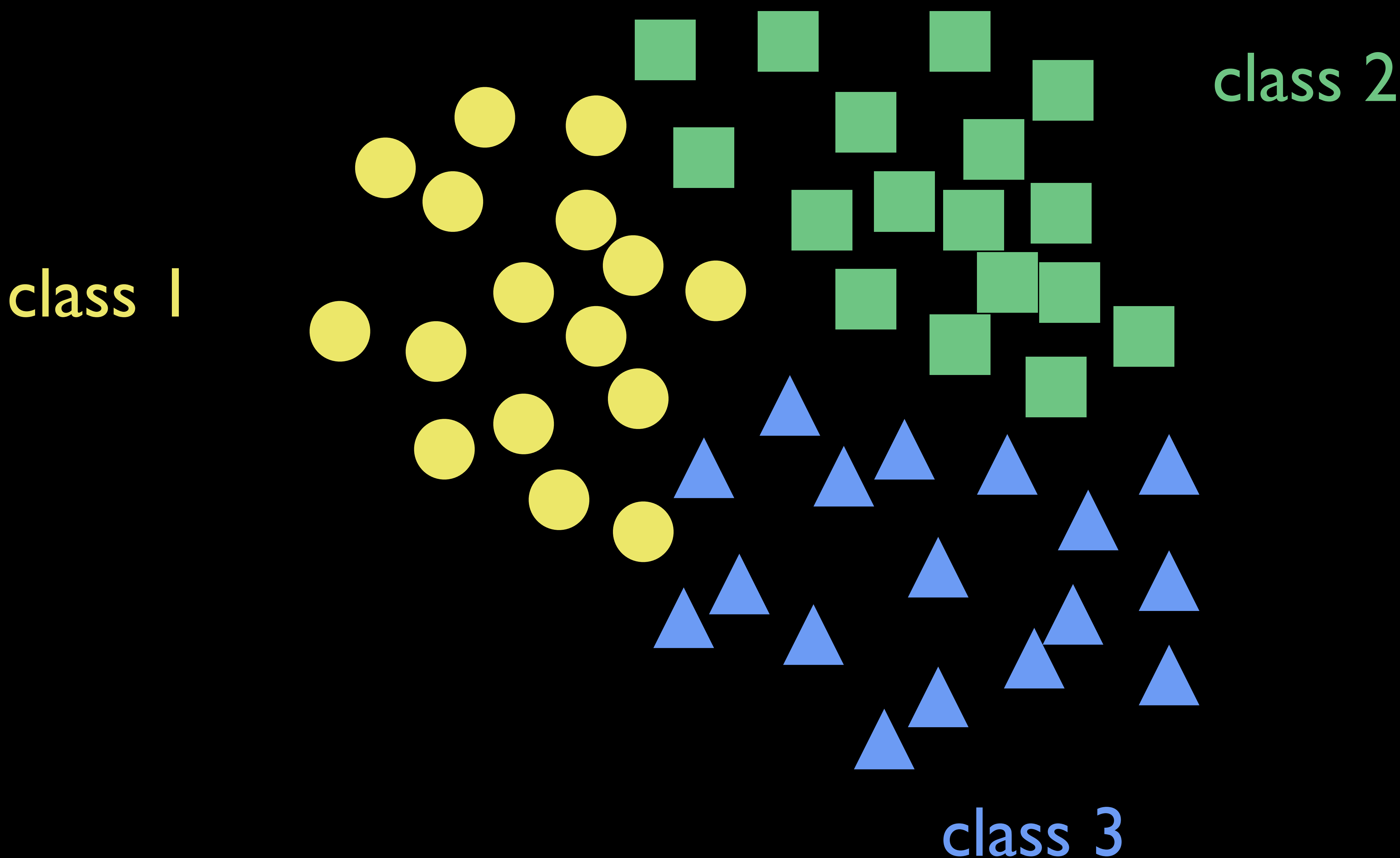
Stochastic Neighbor Compression

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

Washington University in St. Louis

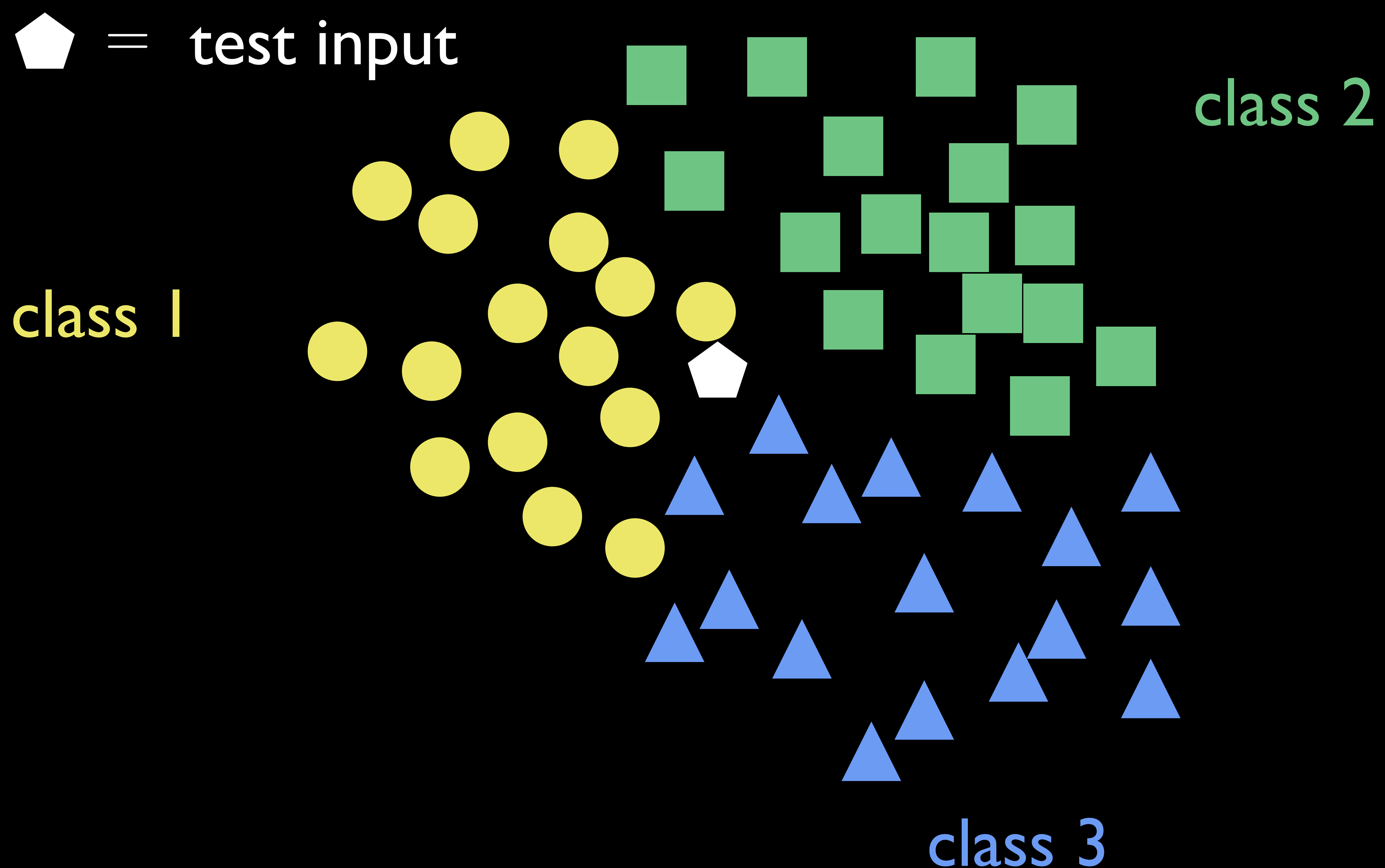
NN Classifier

[Cover & Hart, 1967]



NN Classifier

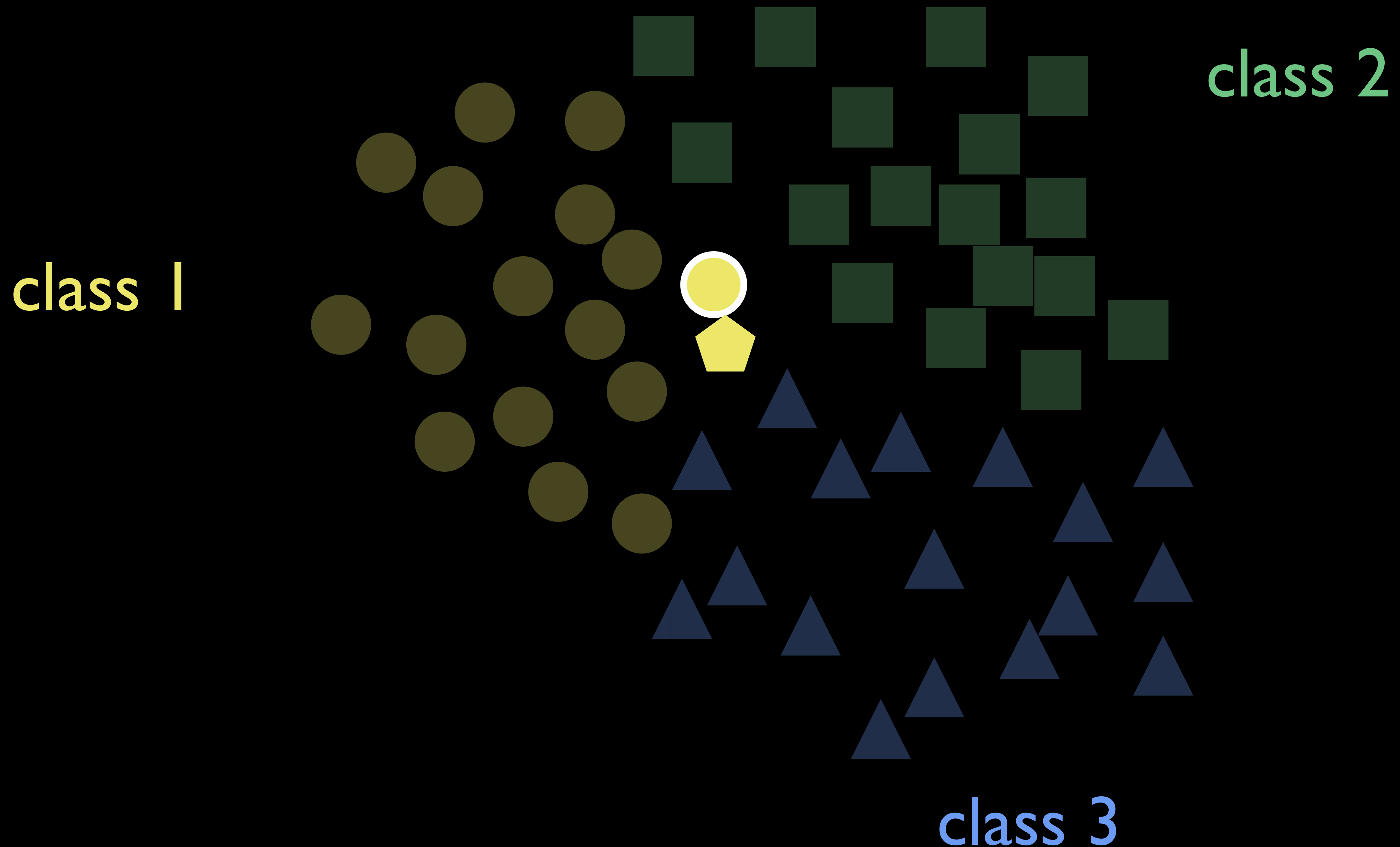
[Cover & Hart, 1967]



NN Classifier

[Cover & Hart, 1967]

1-nearest neighbor rule

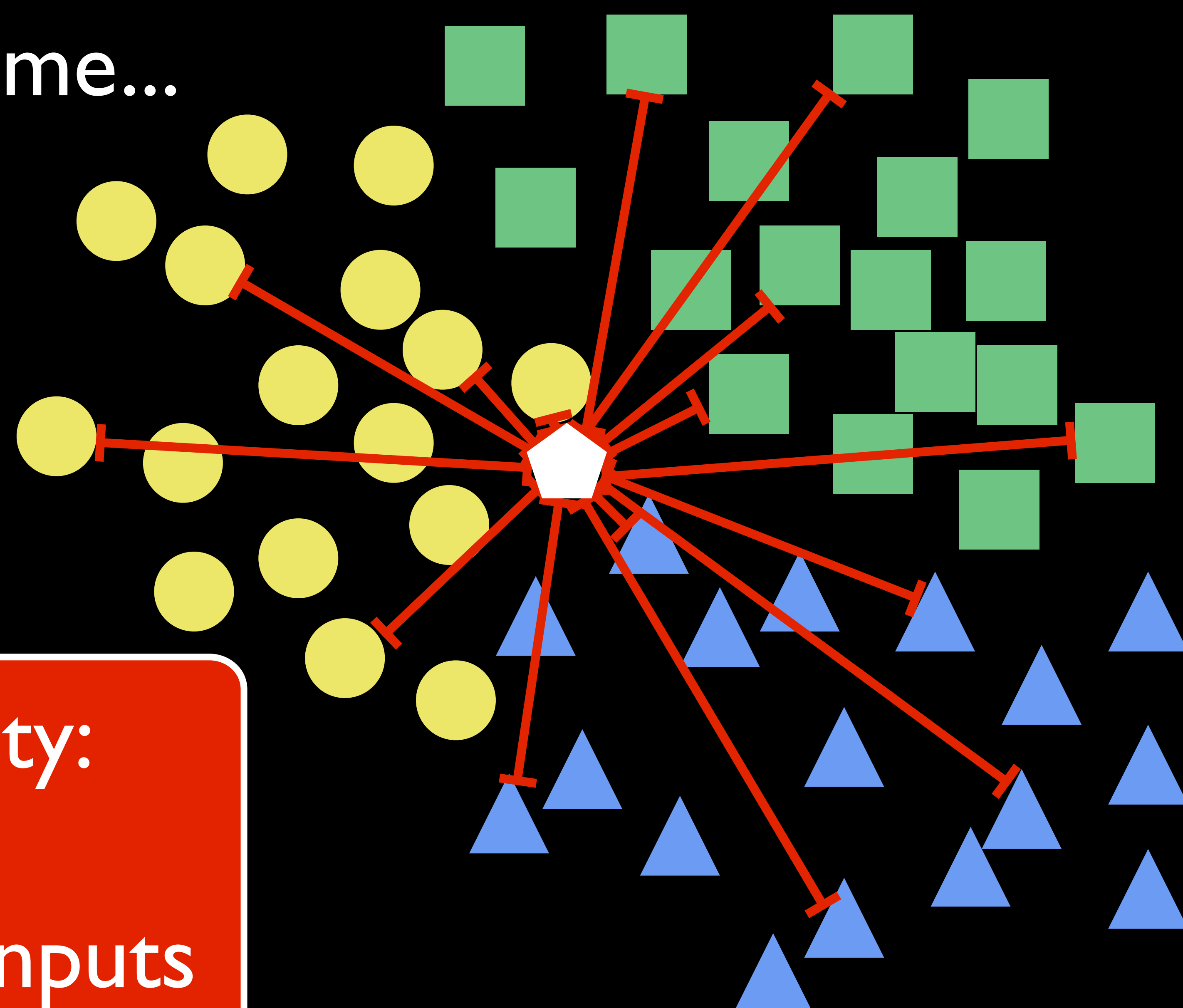


NN Classifier

[Cover & Hart, 1967]

1-nearest neighbor rule

during test-time...



complexity:

$\mathrm{O}(dn)$

n training inputs

d features

NN Classifier

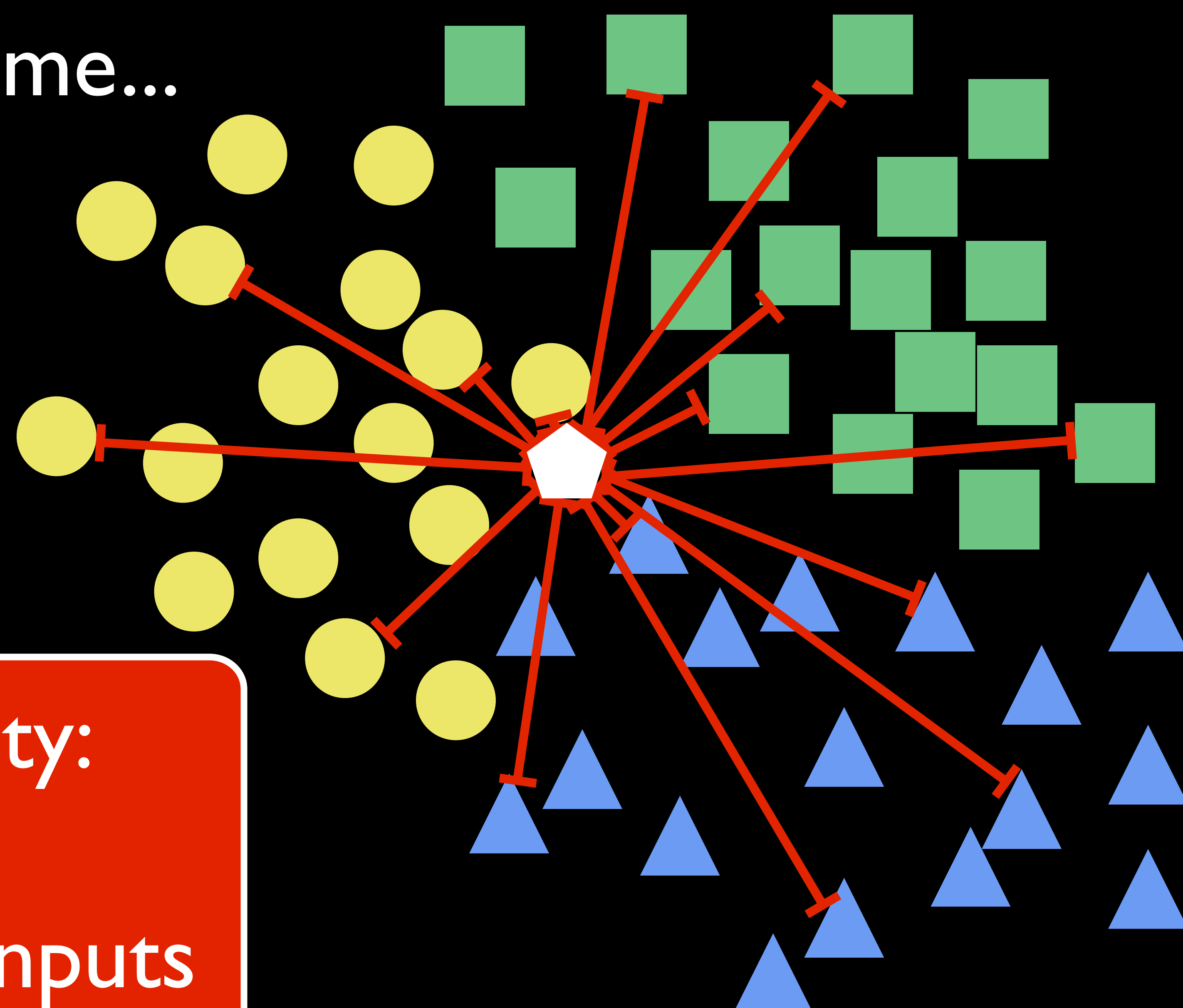
[Cover & Hart, 1967]

1-nearest neighbor rule

during test-time...

class 1

class 2



class 3

for each test input!

complexity:

$\mathrm{O}(dn)$

n training inputs

d features

e.g.

$n = 6 \times 10^4$

$d = 784$

$\mathrm{O}(dn)$

≈ 47 million

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:

$$\mathrm{O}(dn)$$

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

<u>Training Consistent Sampling</u>

[Hart, 1968]

[Anguilli, 2005]

<u>Prototype Generation</u>

[Bandyopadhyay & Maulik, 2002]

[Mollineda et al., 2002]

<u>Prototype Positioning</u>

[Bermejo & Cabestany, 1999]

[Toussaint 2002]

NN Classifier

1-nearest neighbor rule

complexity:

$$\mathrm{O}\left(dn\right)$$

n training inputs

d features

How can we reduce this complexity?

1. reduce n
2. reduce d

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

<u>Dimensionality Reduction</u>

[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

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

<u>Tree Structures</u>

[Omohundro, 1989]

[Beygelzimer et al., 2006]

<u>Hashing</u>

[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?

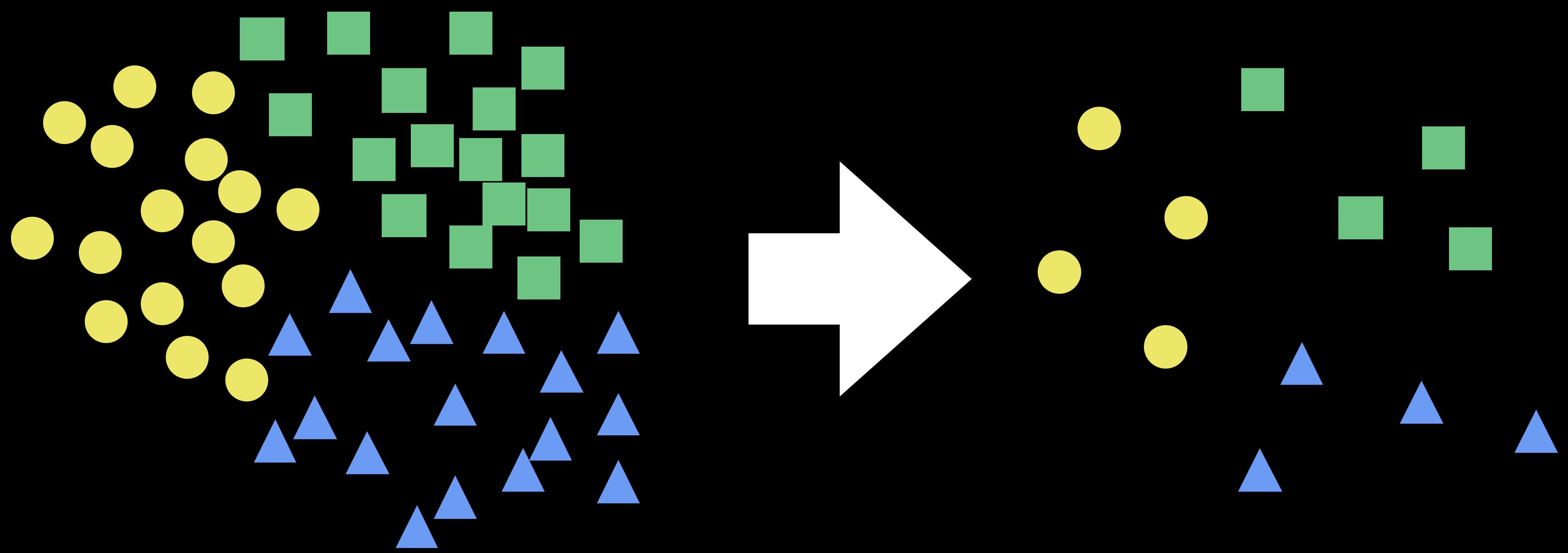
1. **reduce** n
2. reduce d
3. use data structures

Dataset Compression

Main Idea

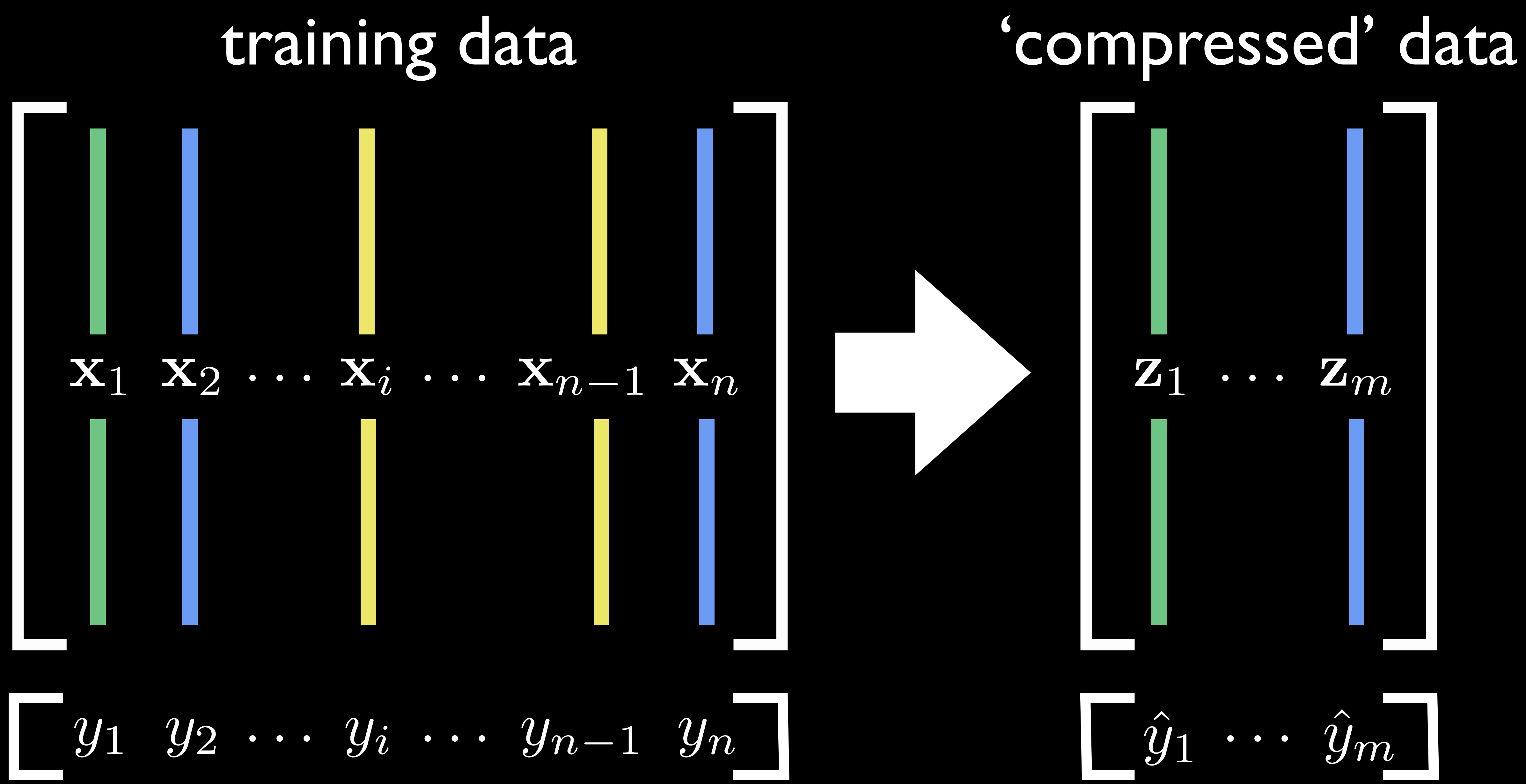
learn new synthetic inputs!

training data ‘compressed’ data



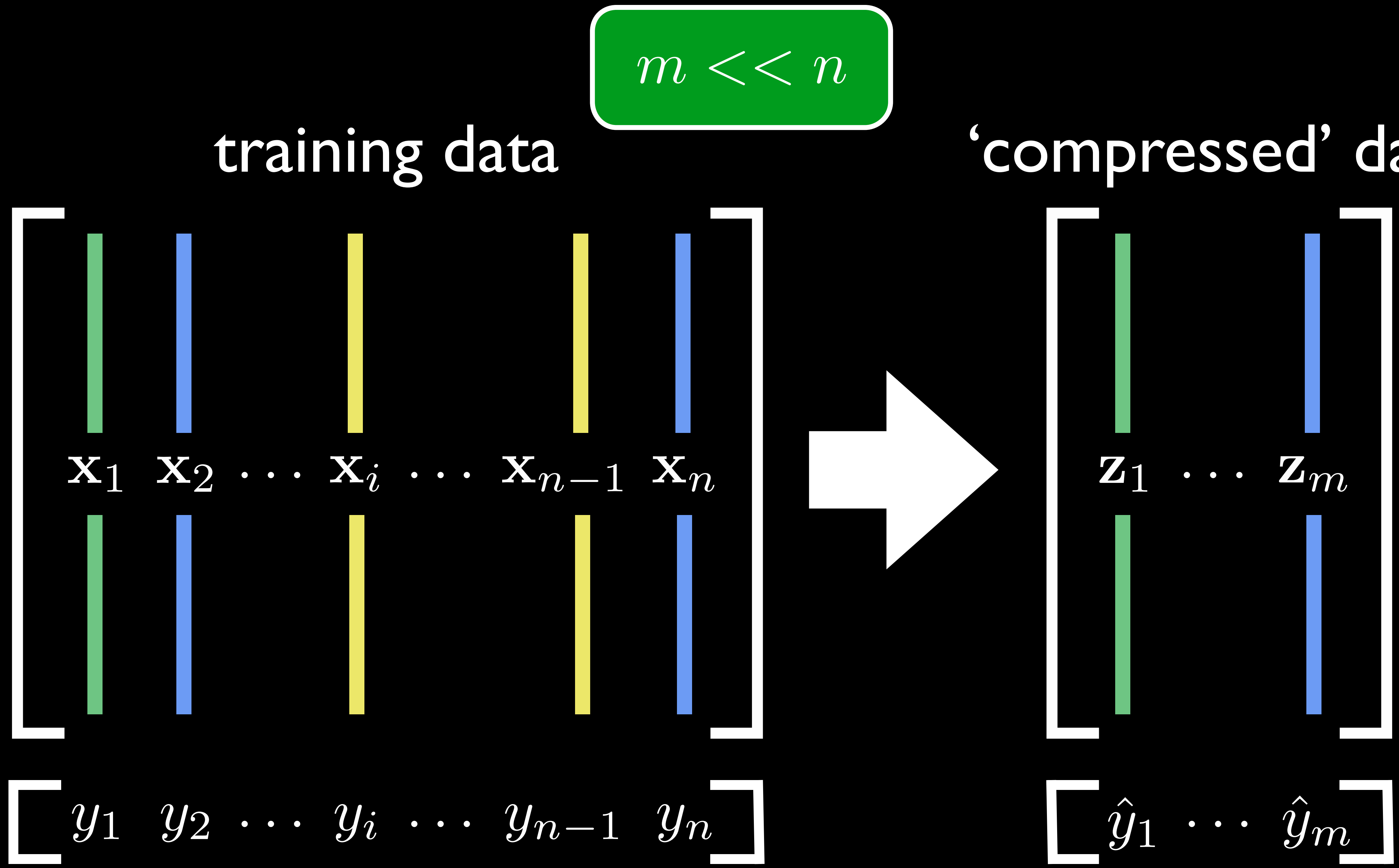
Main Idea

learn new synthetic inputs!



Main Idea

learn new synthetic inputs!



Main Idea

learn new synthetic inputs!

use ‘compressed’ set to make future predictions

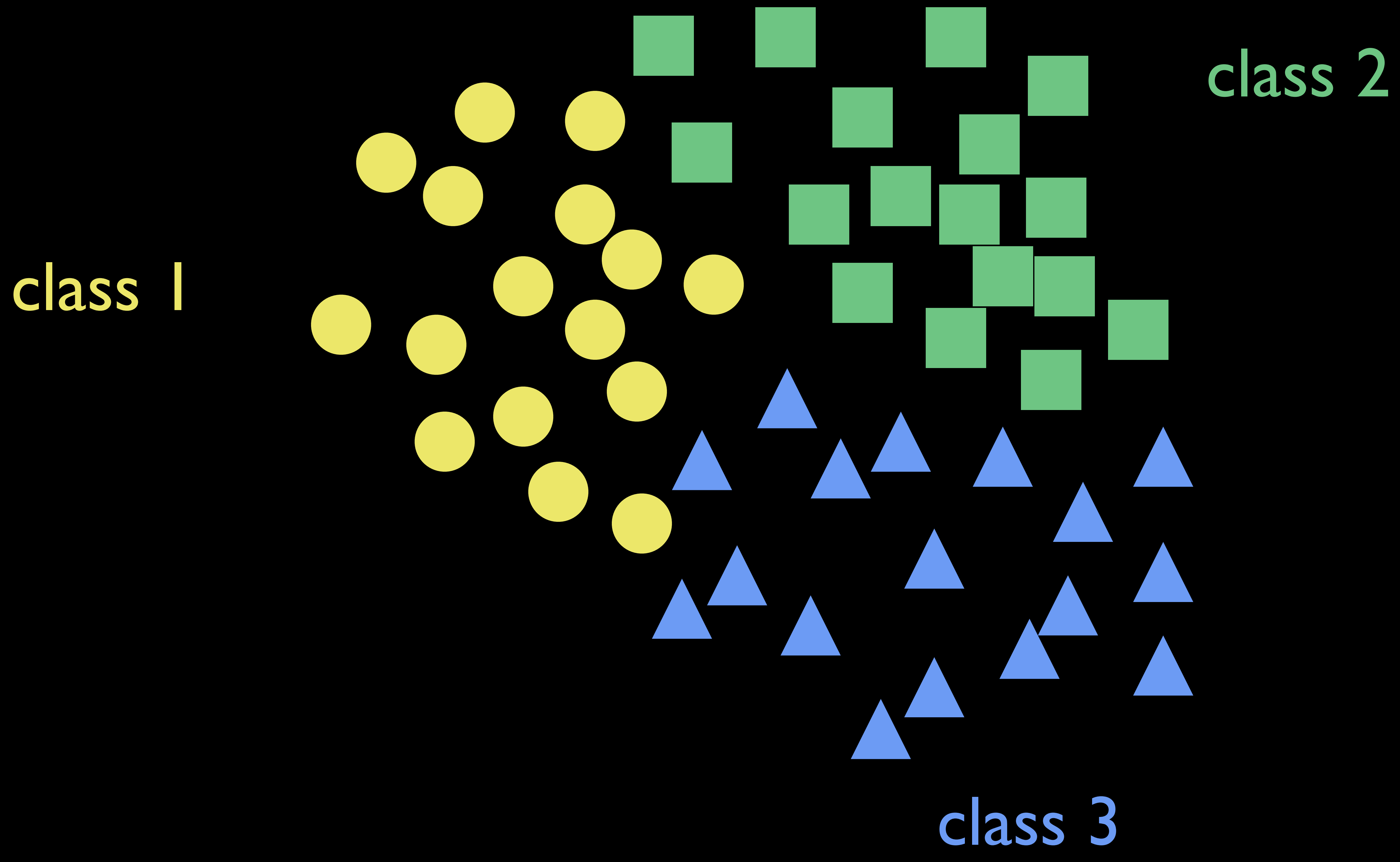
‘compressed’ data

$$\begin{bmatrix} \mathbf{z}_1 & \cdots & \mathbf{z}_m \end{bmatrix}$$

$$\begin{bmatrix} \hat{y}_1 & \cdots & \hat{y}_m \end{bmatrix}$$

Approach

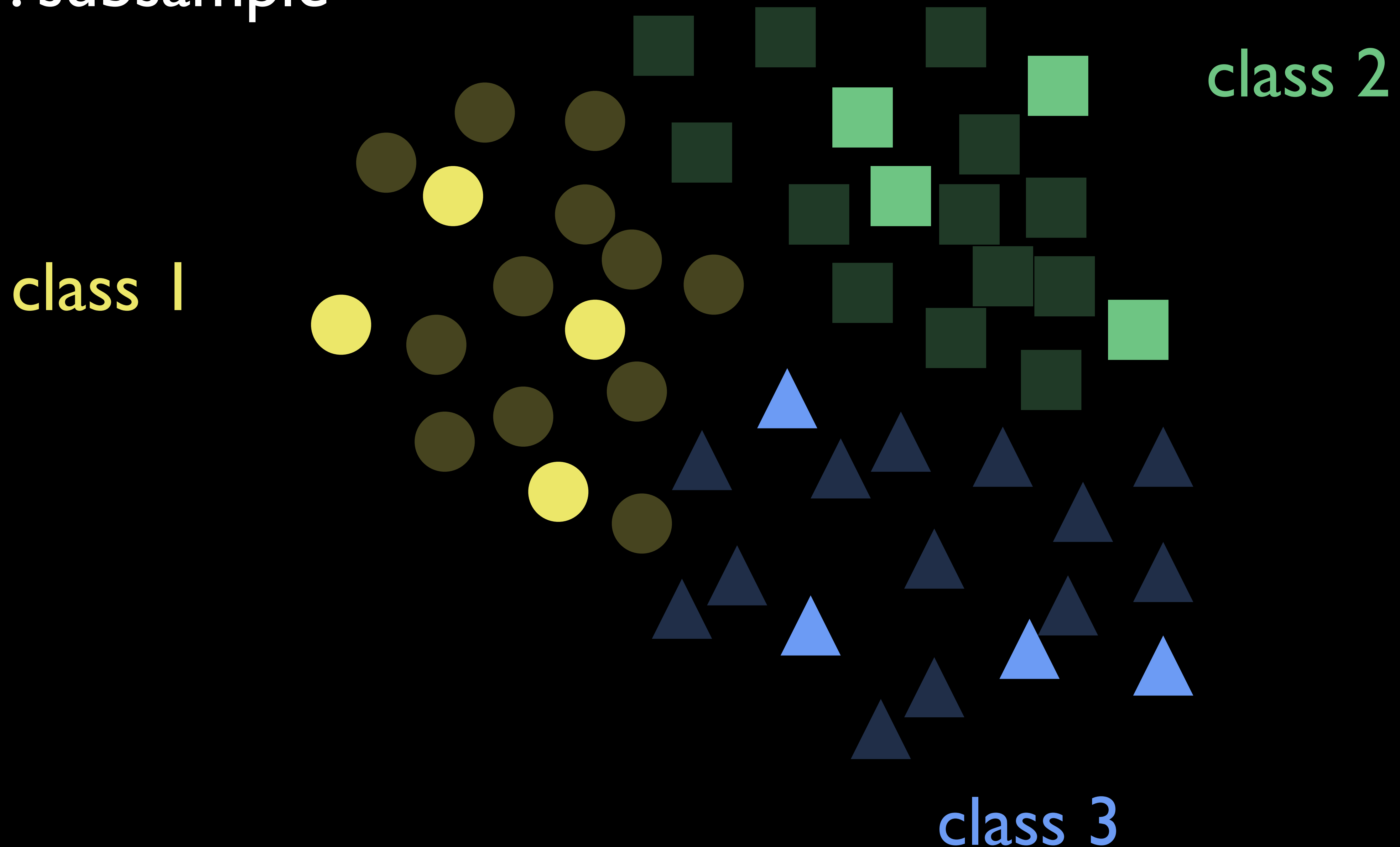
steps to a new training set:



Approach

steps to a new training set:

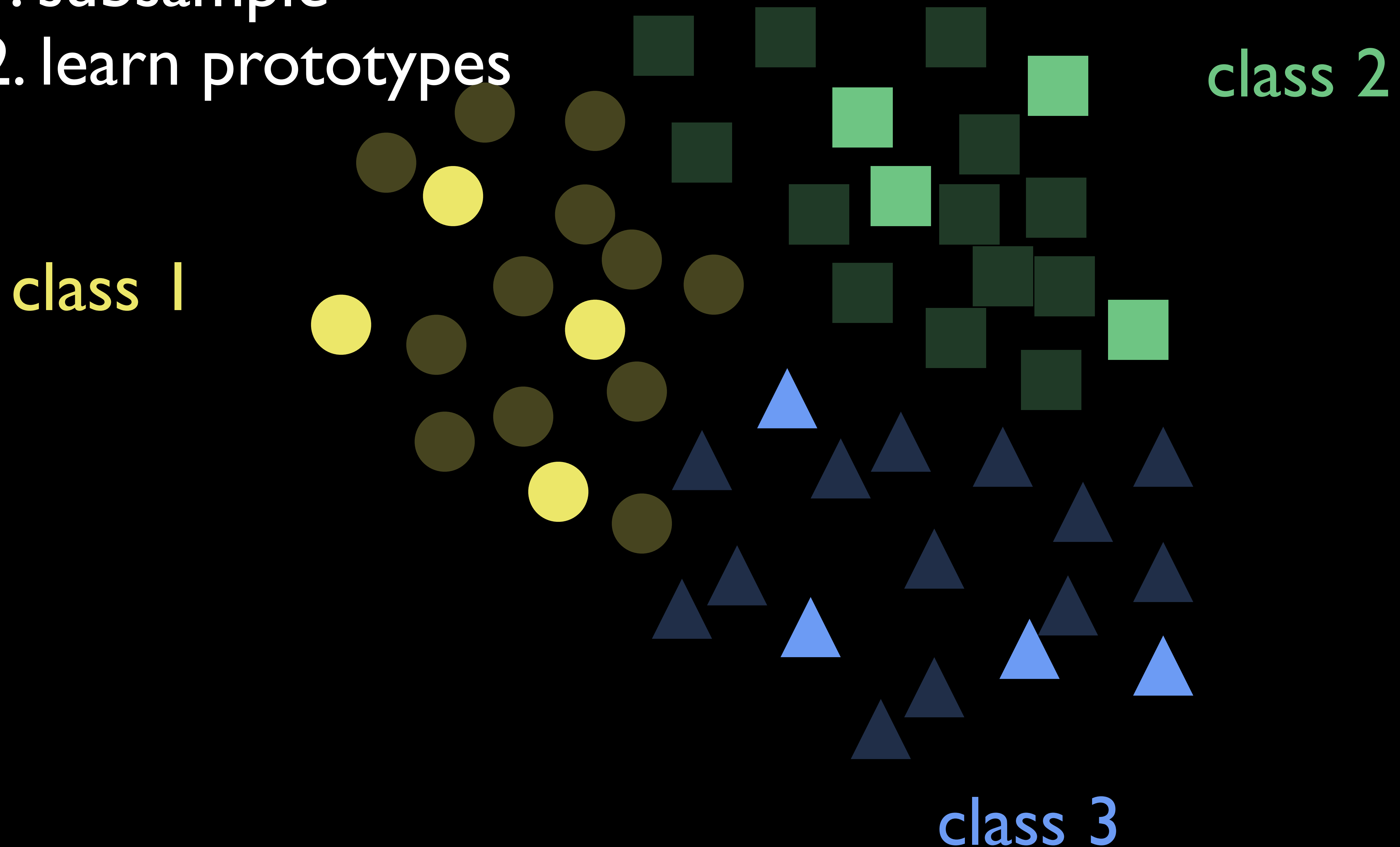
1. subsample



Approach

steps to a new training set:

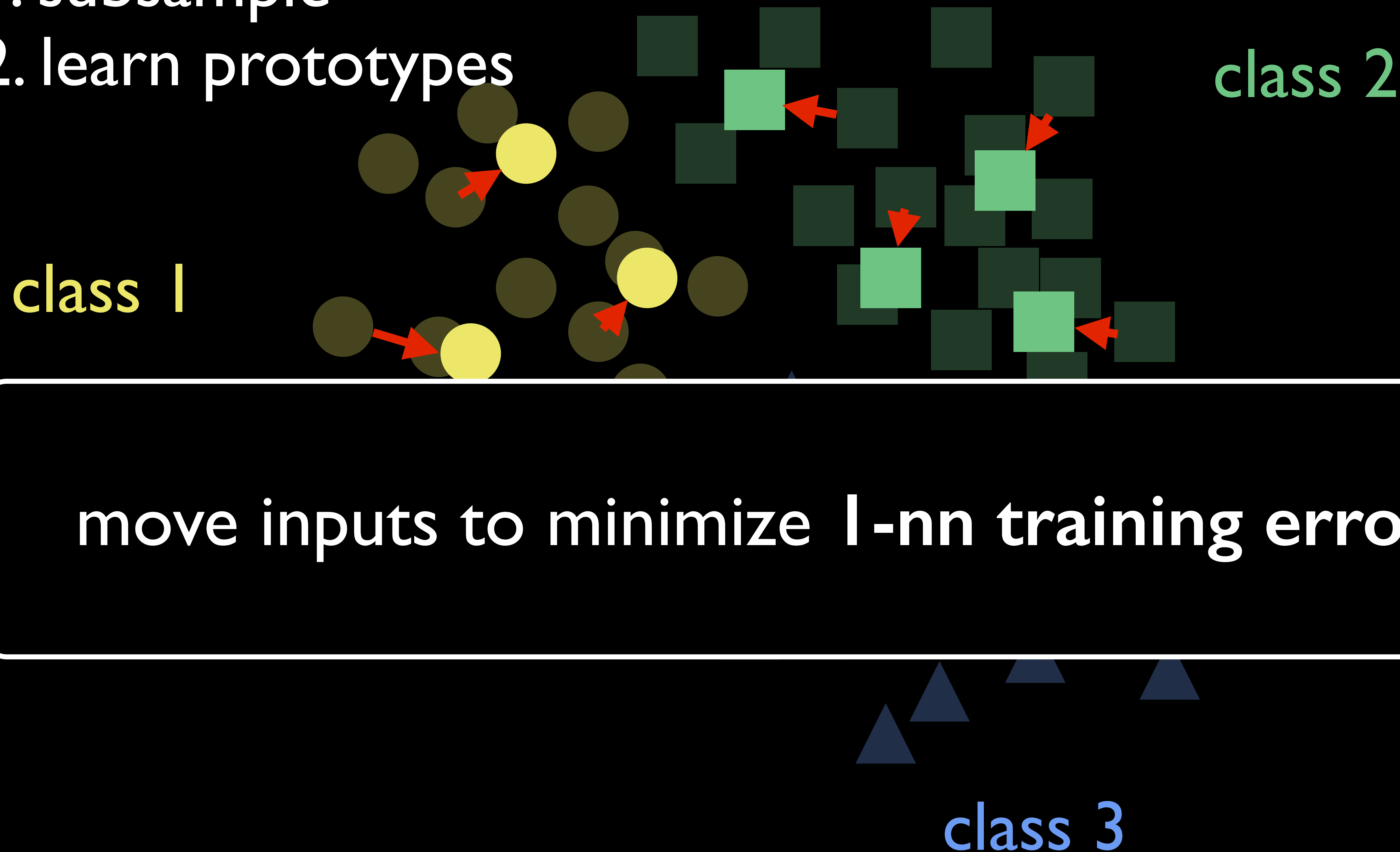
1. subsample
2. learn prototypes



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

$$\min_{\mathbf{z}_1,\ldots,\mathbf{z}_m} \sum_{i=1}^{n} [\mathrm{nn}_{\{\mathbf{z}_1,\ldots,\mathbf{z}_m\}}(\mathbf{x}_i) \neq y_i]$$

label of nearest neighbor to $\mathbf{x}_i$ in set $\{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$

true label

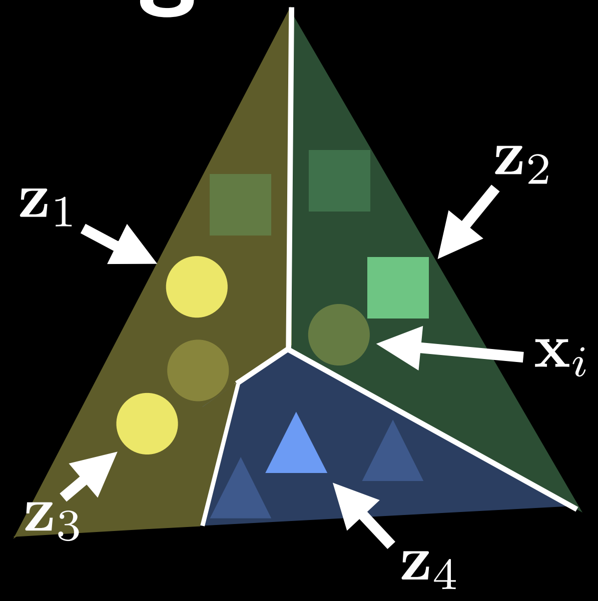
Learning Prototypes

move inputs to minimize 1-nn training error

$$\min_{\mathbf{z}_1,\ldots,\mathbf{z}_m} \sum_{i=1}^{n} [\mathrm{nn}_{\{\mathbf{z}_1,\ldots,\mathbf{z}_m\}}(\mathbf{x}_i) \neq y_i]$$

label of nearest neighbor to $\mathbf{x}_i$ in set $\{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$

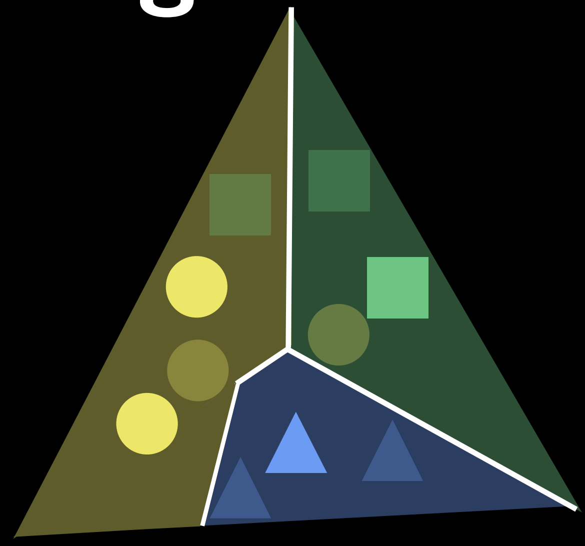
true label



Learning Prototypes

move inputs to minimize 1-nn training error

$$\cancel{\min_{\mathbf{z}_1,\dots,\mathbf{z}_m} \sum_{i=1}^{n} [\mathrm{nn}_{\{\mathbf{z}_1,\dots,\mathbf{z}_m\}}(\mathbf{x}_i) \neq y_i]}$$

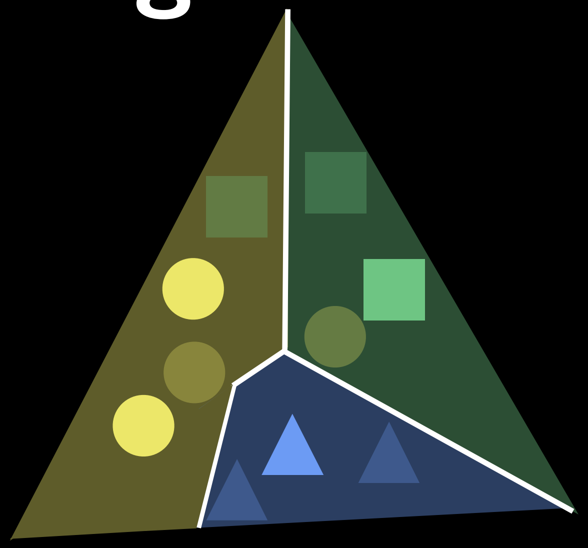


not continous
not differentiable

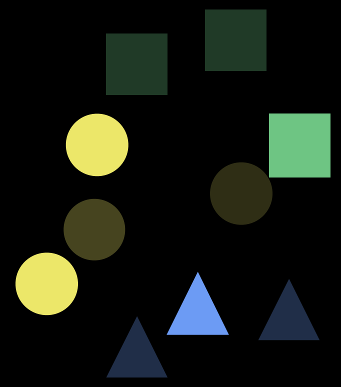
Learning Prototypes

move inputs to minimize 1-nn training error

$$\xcancel{\min_{\mathbf{z}_1,\ldots,\mathbf{z}_m} \sum_{i=1}^{n} [\mathrm{nn}_{\{\mathbf{z}_1,\ldots,\mathbf{z}_m\}}(\mathbf{x}_i) \neq y_i]}$$



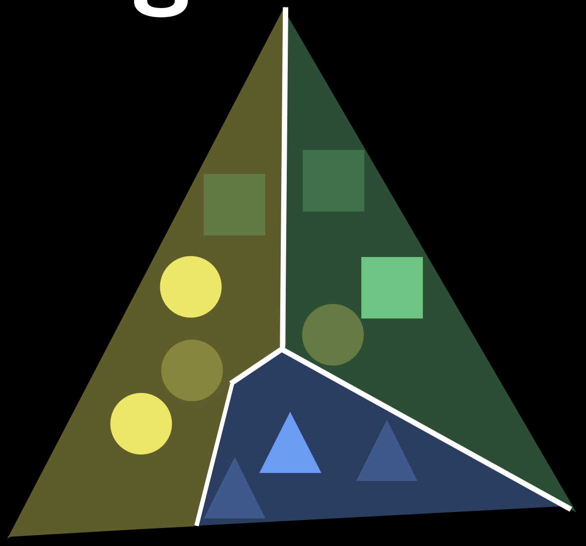
Stochastic Neighborhood
[Hinton & Roweis, 2002]



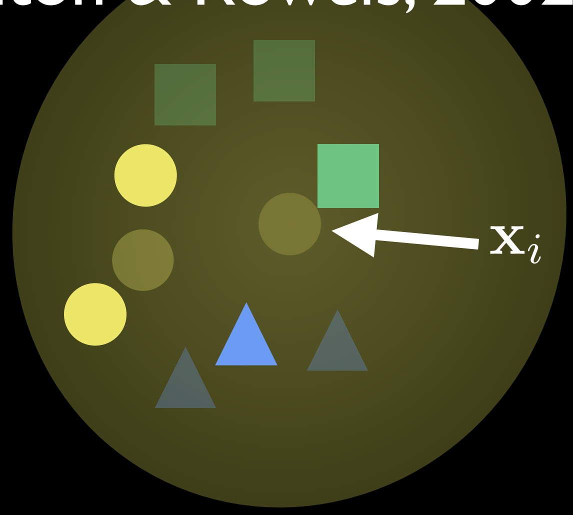
Learning Prototypes

move inputs to minimize 1-nn training error

$$\cancel{\min_{\mathbf{z}_1,\ldots,\mathbf{z}_m} \sum_{i=1}^{n} [\mathrm{nn}_{\{\mathbf{z}_1,\ldots,\mathbf{z}_m\}}(\mathbf{x}_i) \neq y_i]}$$



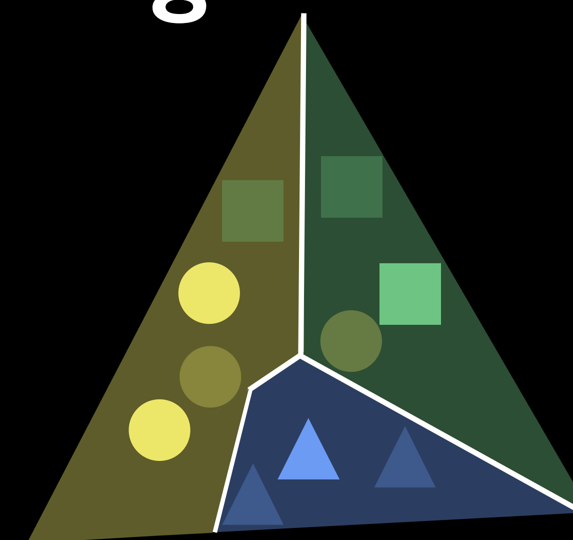
Stochastic Neighborhood
[Hinton & Roweis, 2002]



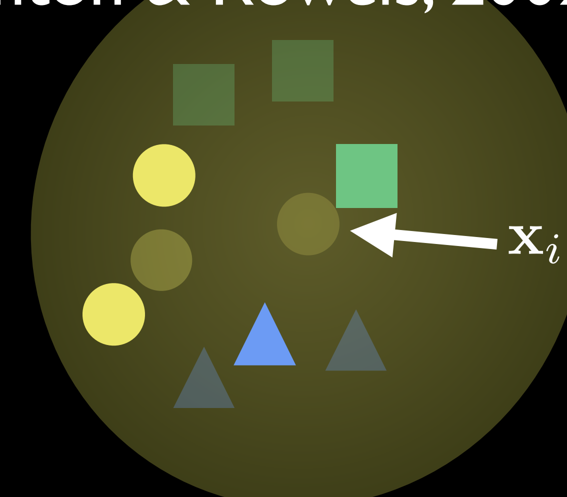
Learning Prototypes

move inputs to minimize 1-nn training error

$$\cancel{\min_{\mathbf{z}_1,\ldots,\mathbf{z}_m} \sum_{i=1}^{n} [\mathrm{nn}_{\{\mathbf{z}_1,\ldots,\mathbf{z}_m\}}(\mathbf{x}_i) \neq y_i]}$$



Stochastic Neighborhood
[Hinton & Roweis, 2002]



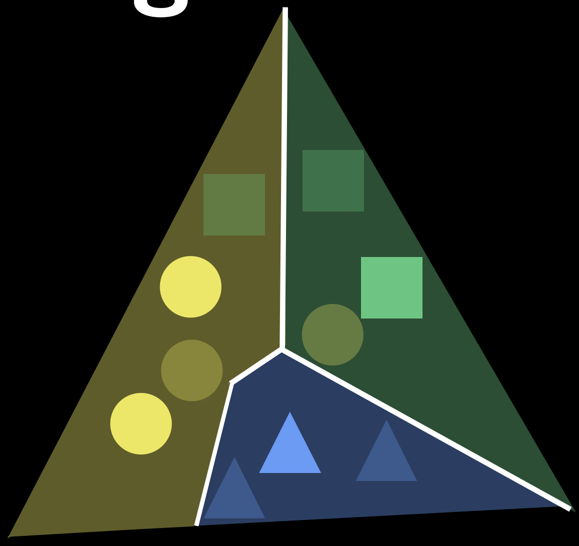
p_i

probability $\mathbf{x}_i$ is predicted correctly

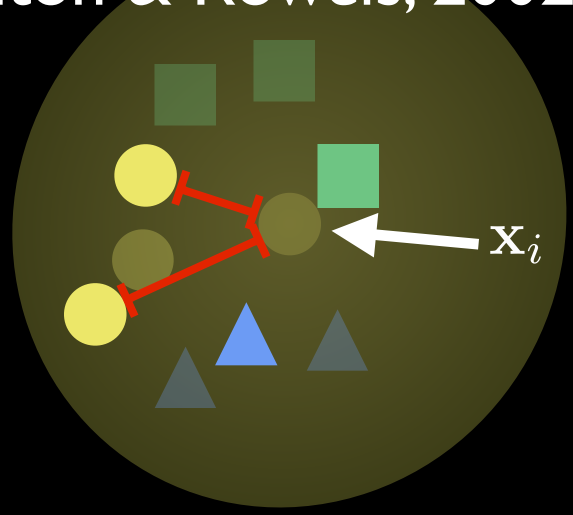
Learning Prototypes

move inputs to minimize 1-nn training error

$$\cancel{\min_{\mathbf{z}_1,\ldots,\mathbf{z}_m} \sum_{i=1}^{n} [\text{nn}_{\{\mathbf{z}_1,\ldots,\mathbf{z}_m\}}(\mathbf{x}_i) \neq y_i]}$$



Stochastic Neighborhood
[Hinton & Roweis, 2002]



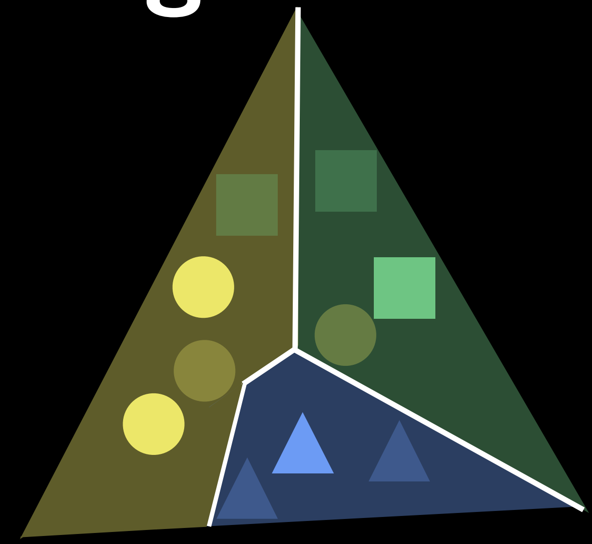
p_i

probability $\mathbf{x}_i$ is predicted correctly $\triangleq \frac{1}{Z} \sum_{j:\hat{y}_j = y_i} \exp(-\gamma^2 \|\mathbf{x}_i - \mathbf{z}_j\|_2^2)$

Learning Prototypes

move inputs to minimize 1-nn training error

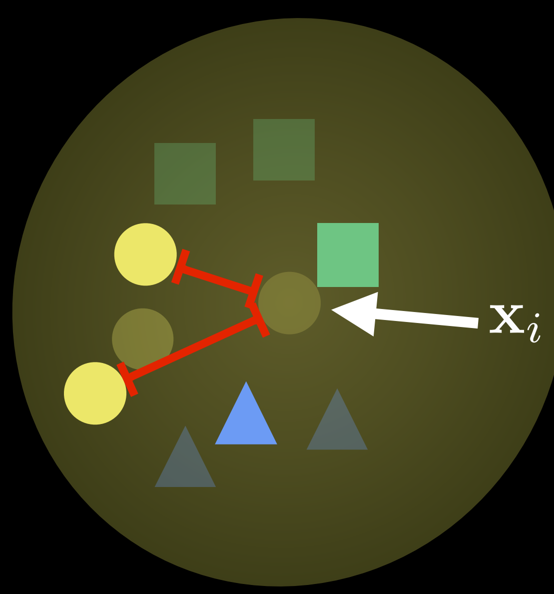
$$\cancel{\min_{\mathbf{z}_1,\dots,\mathbf{z}_m}\sum_{i=1}^{n}[\mathrm{nn}_{\{\mathbf{z}_1,\dots,\mathbf{z}_m\}}(\mathbf{x}_i)\neq y_i]}$$



minimize negative log likelihood

$$\min_{\mathbf{z}_1,\dots,\mathbf{z}_m}\sum_{i=1}^{n}-\log(p_i)$$

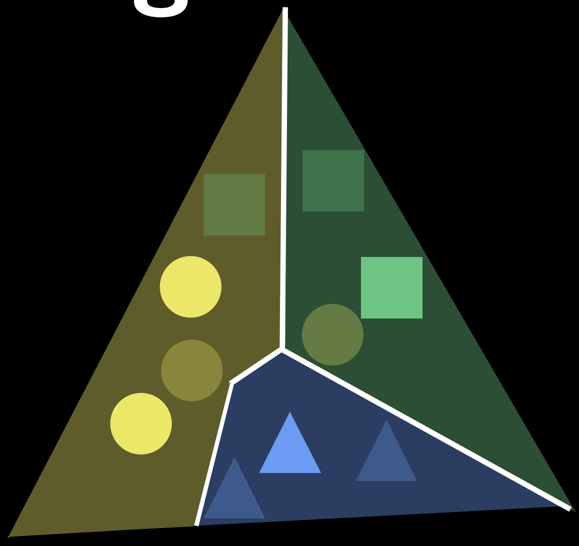
probability $\mathbf{x}_i$ is predicted correctly $\triangleq \frac{1}{Z}\sum_{j:\hat{y}_j=y_i}\exp(-\gamma^2\|\mathbf{x}_i-\mathbf{z}_j\|_2^2)$



Learning Prototypes

move inputs to minimize 1-nn training error

$$\cancel{\min_{\mathbf{z}_1,\dots,\mathbf{z}_m} \sum_{i=1}^{n} [\mathrm{nn}_{\{\mathbf{z}_1,\dots,\mathbf{z}_m\}}(\mathbf{x}_i) \neq y_i]}$$

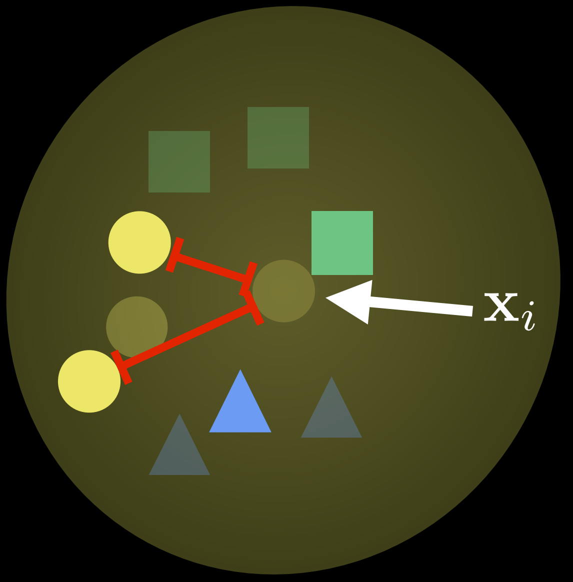


use conjugate gradient descent!

minimize **negative log likelihood**

$$\min_{\mathbf{z}_1,\dots,\mathbf{z}_m} \sum_{i=1}^{n} -\log(p_i)$$

probability $\mathbf{x}_i$ is predicted correctly $\triangleq \frac{1}{Z} \sum_{j:\hat{y}_j = y_i} \exp(-\gamma^2 \|\mathbf{x}_i - \mathbf{z}_j\|_2^2)$



Results

[a motivating visualization]

Results

Yale-Faces

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

Results

Yale-Faces

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

Results

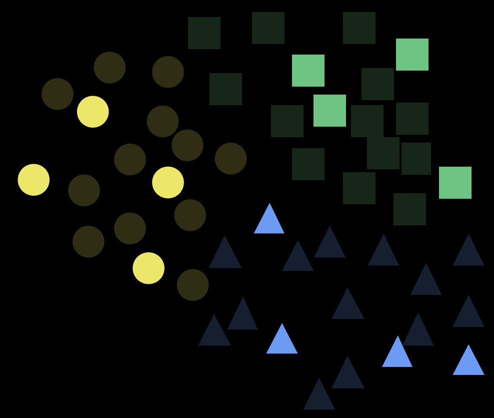
Yale-Faces

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

Results

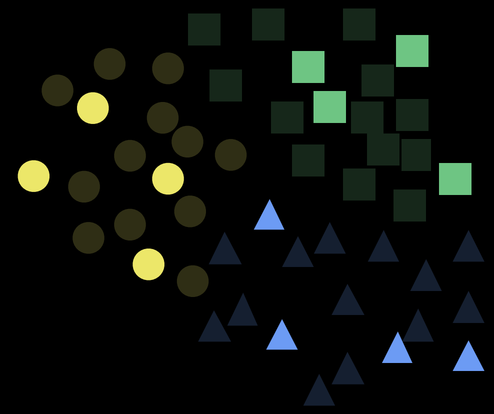
Results

subsampled real faces



Results

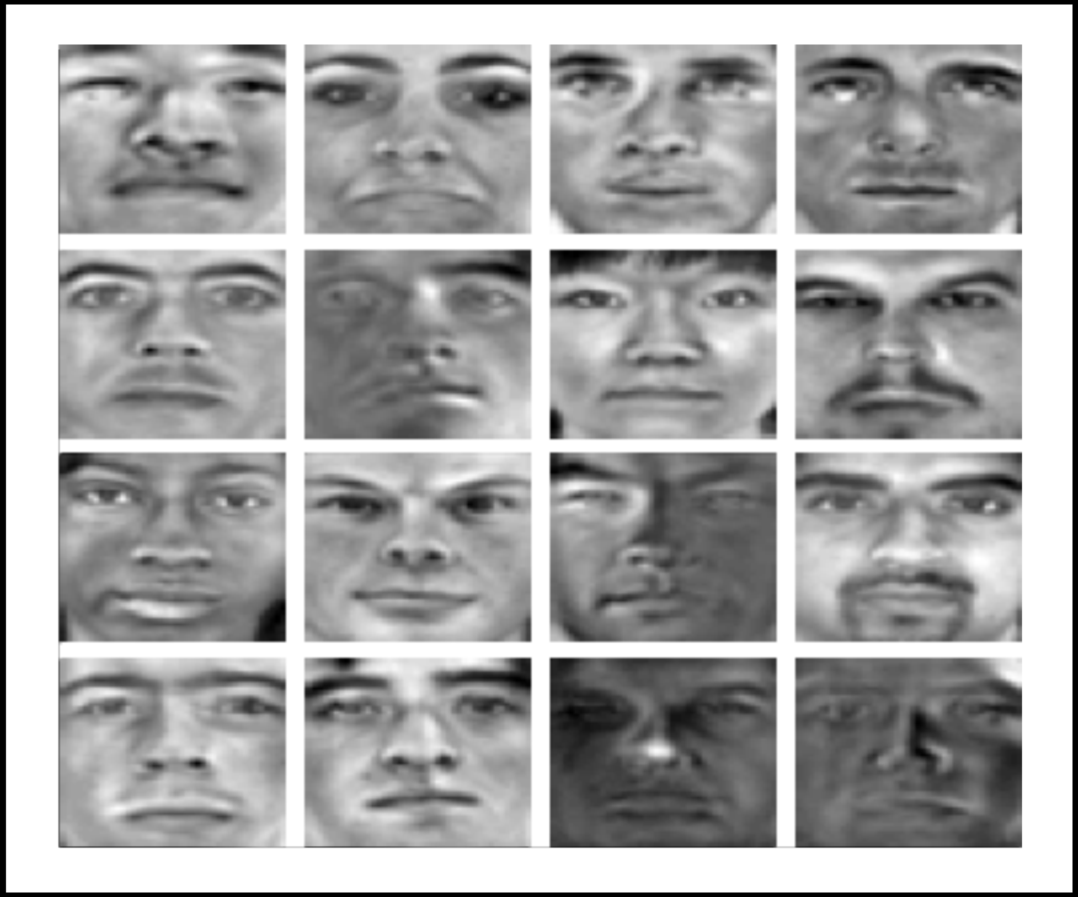
subsampled real faces

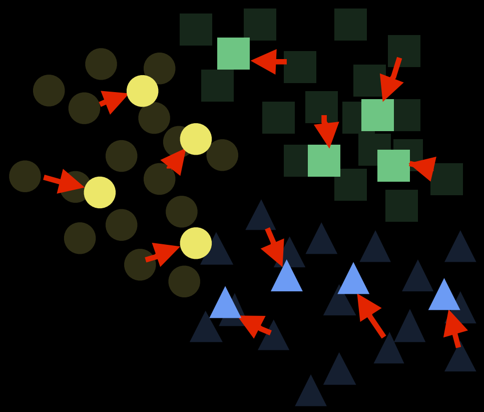


Results

Results

learned faces





Results
[datasets]

Results

Table 1. Characteristics of datasets used in evaluation.

Dataset Statistics			
NAME	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)

Results

training inputs

Table 1. Characteristics of datasets used in evaluation.

DATASET STATISTICS			
NAME	n	$\lvert\mathcal{Y}\rvert$	$d\ (d_L)$
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

number of classes

Table 1. Characteristics of datasets used in evaluation.

DATASET STATISTICS			
NAME	n	$\lvert\mathcal{Y}\rvert$	$d\ (d_L)$
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

Table 1. Characteristics of datasets used in evaluation.

DATASET STATISTICS			
NAME	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)

Results

original & reduced features

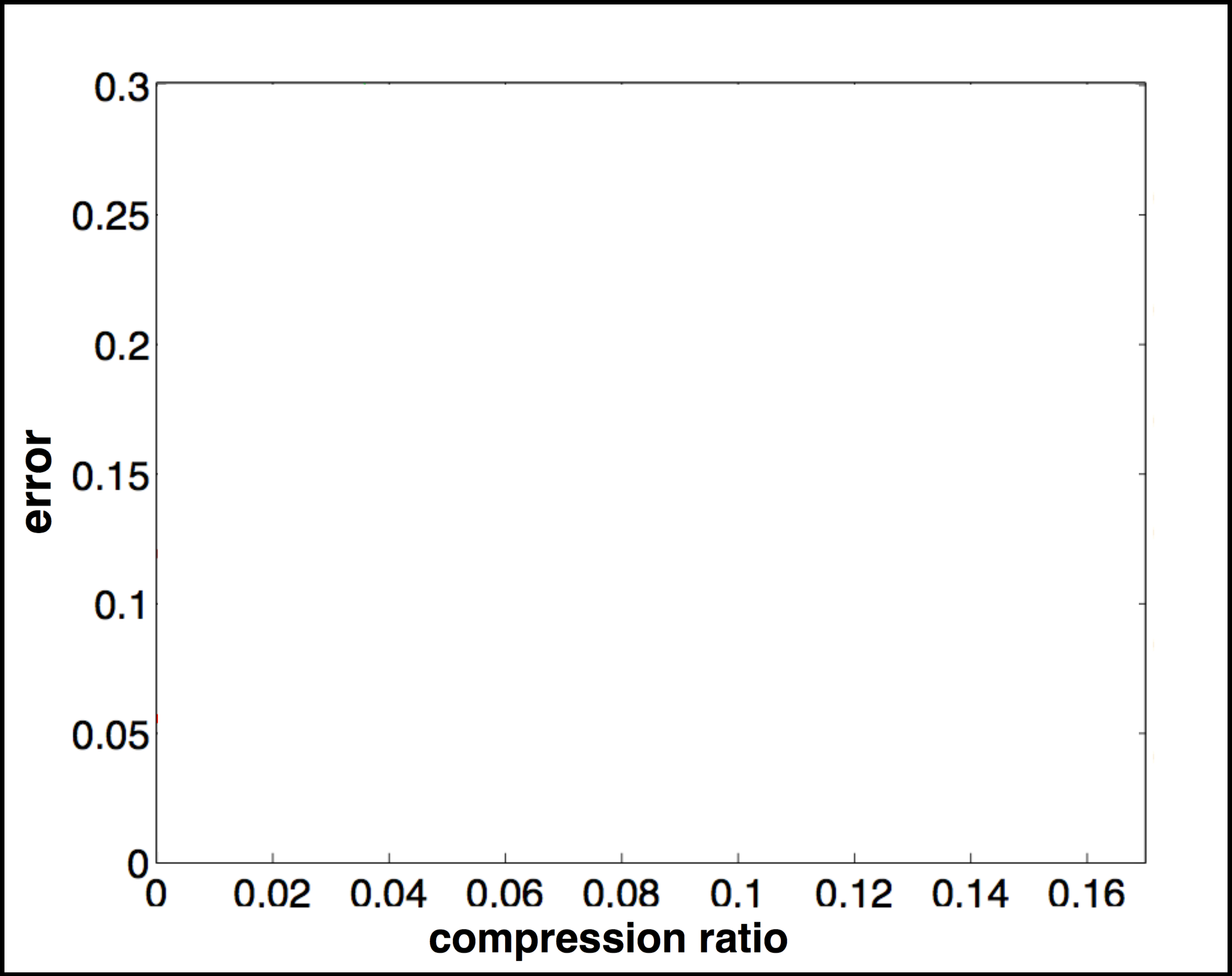
[Weinberger & Saul, 2009]

Table 1. Characteristics of datasets used in evaluation.

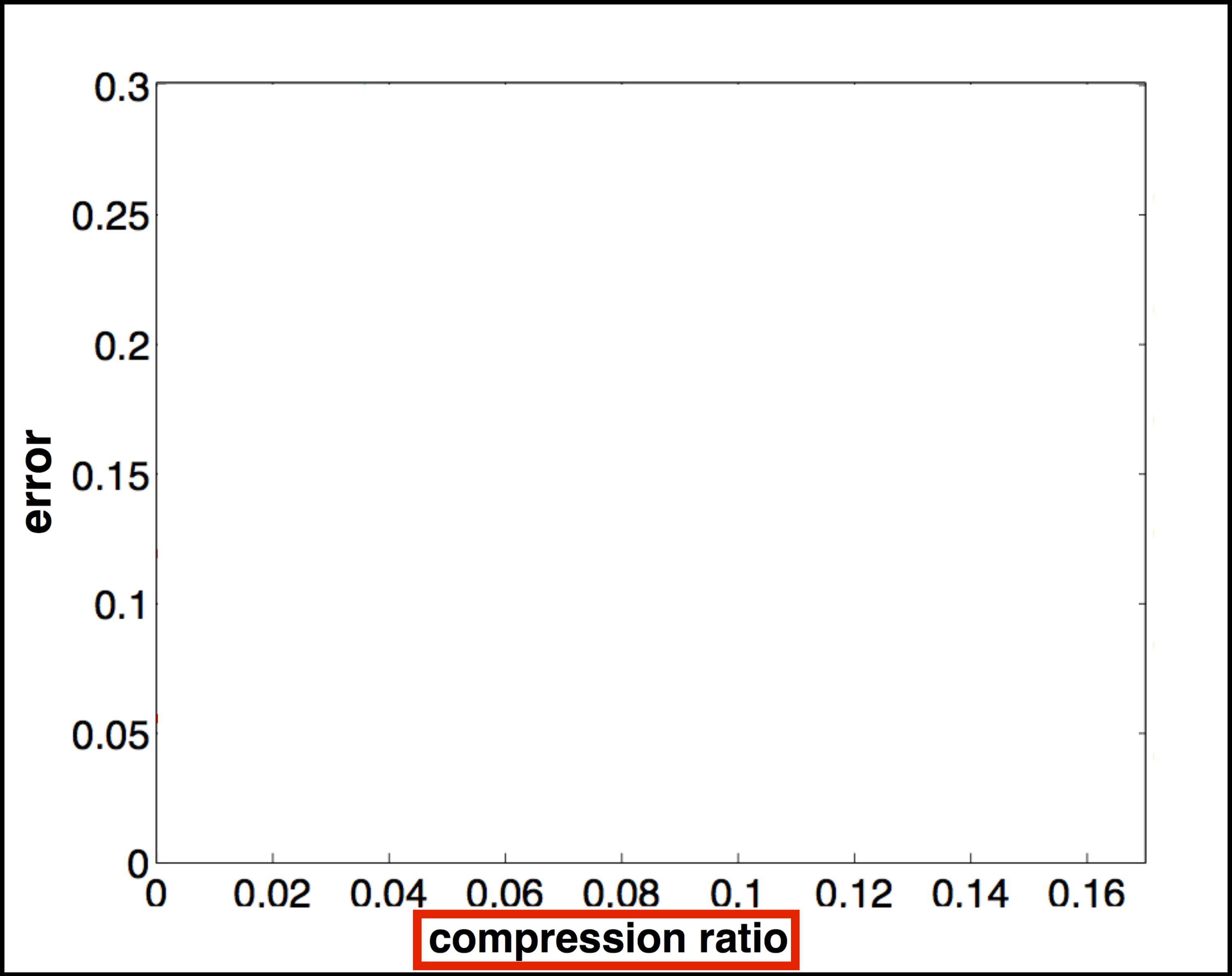
DATASET STATISTICS			
NAME	n	$\lvert\mathcal{Y}\rvert$	$d\ (d_L)$
YALE-FACES	1961	38	8064 (100)
ISOLET	3898	26	617 (172)
LETTERS	16000	26	16 (16)
ADULT	32562	2	123 (50)
W8A	49749	2	300 (100)
MNIST	60000	10	784 (164)
FOREST	100000	7	54 (54)

Results
[error]

Results


0.3
0.25
0.2
0.15
0.1
0.05
0
error
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
compression ratio

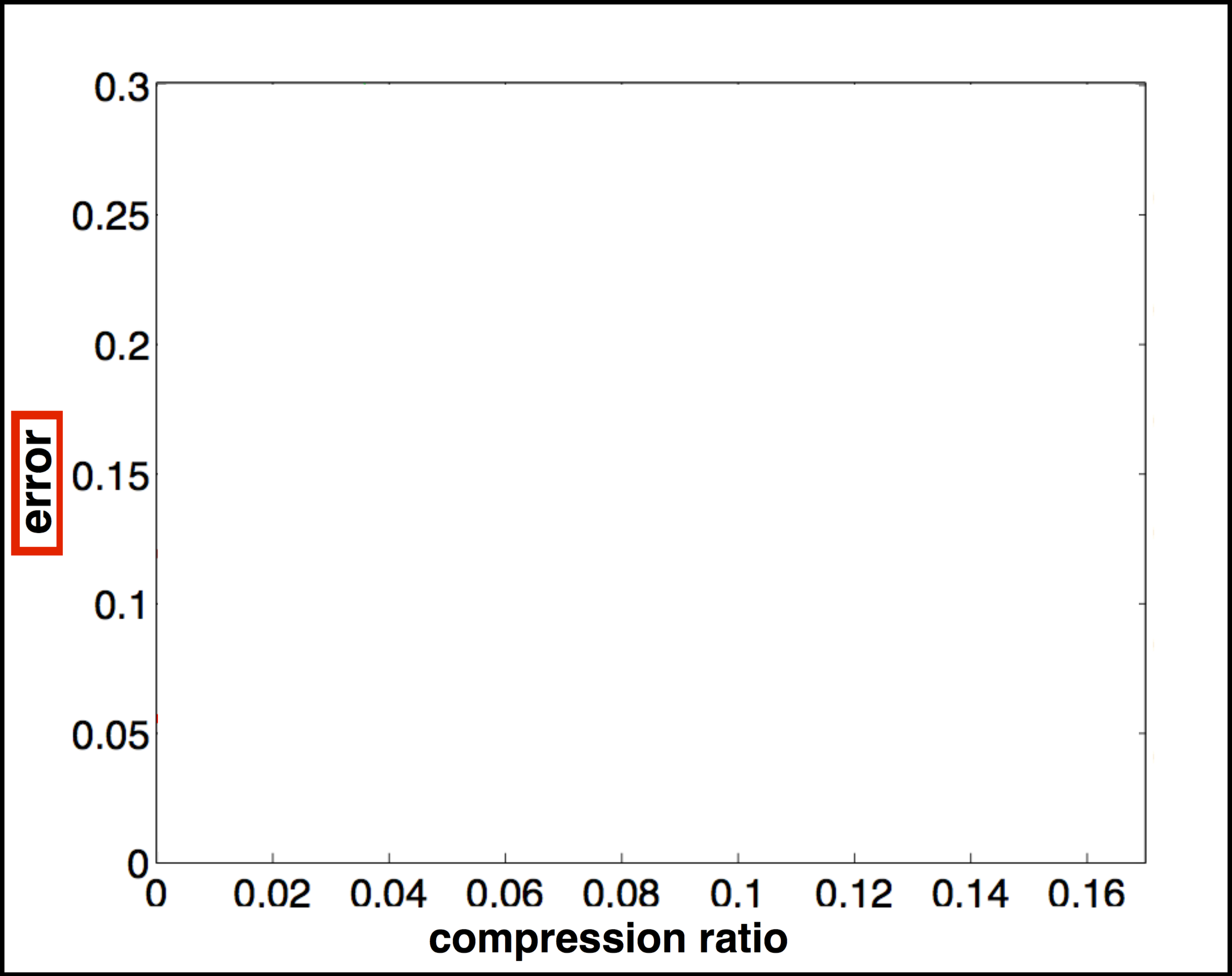
Results



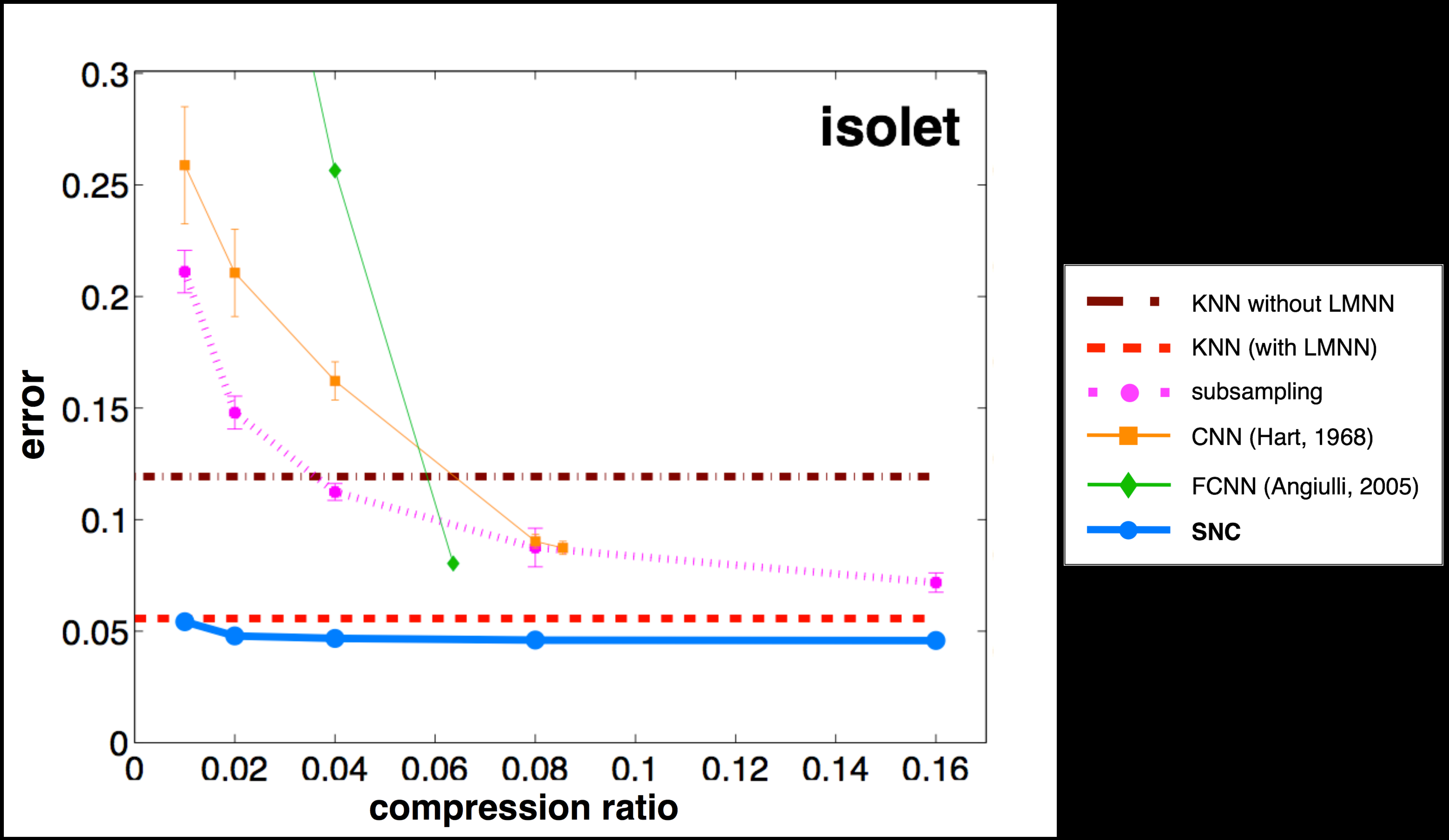
ratio of training inputs in compressed set

Results

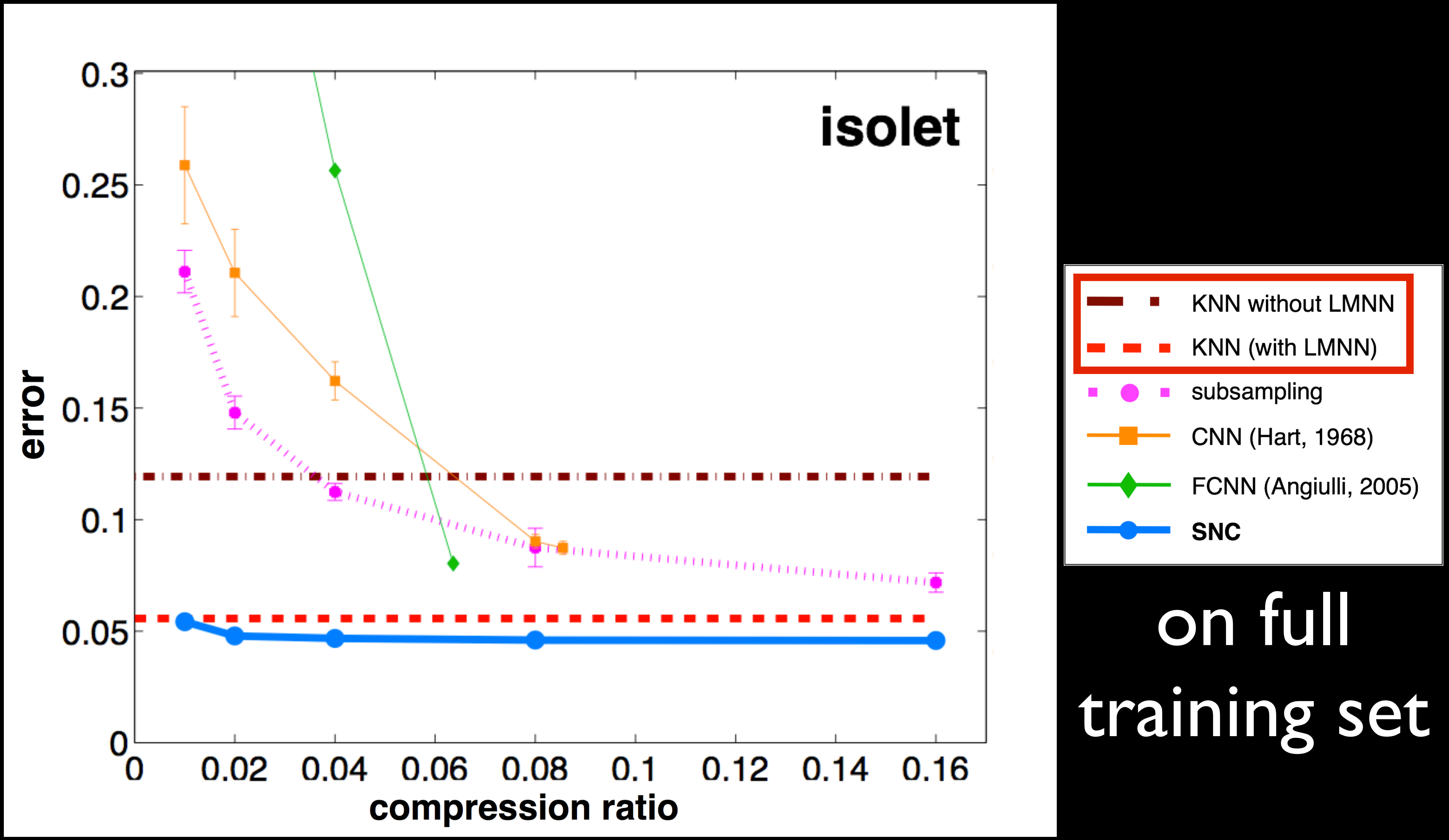
test error



Results

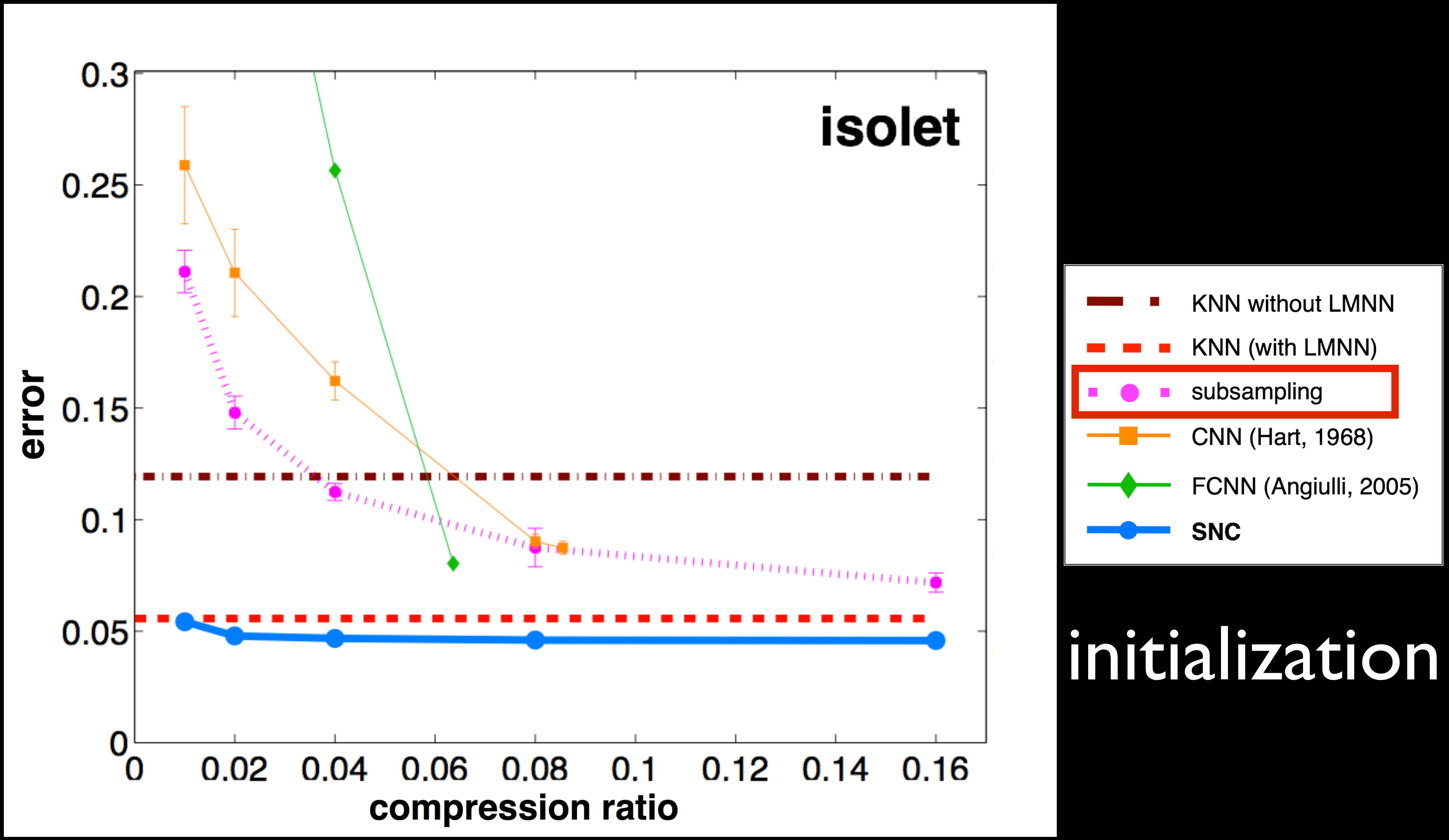

isolet
0.3
0.25
0.2
0.15
0.1
0.05
0
error
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
compression ratio
KNN without LMNN
KNN (with LMNN)
subsampling
CNN (Hart, 1968)
FCNN (Angiulli, 2005)
SNC

Results



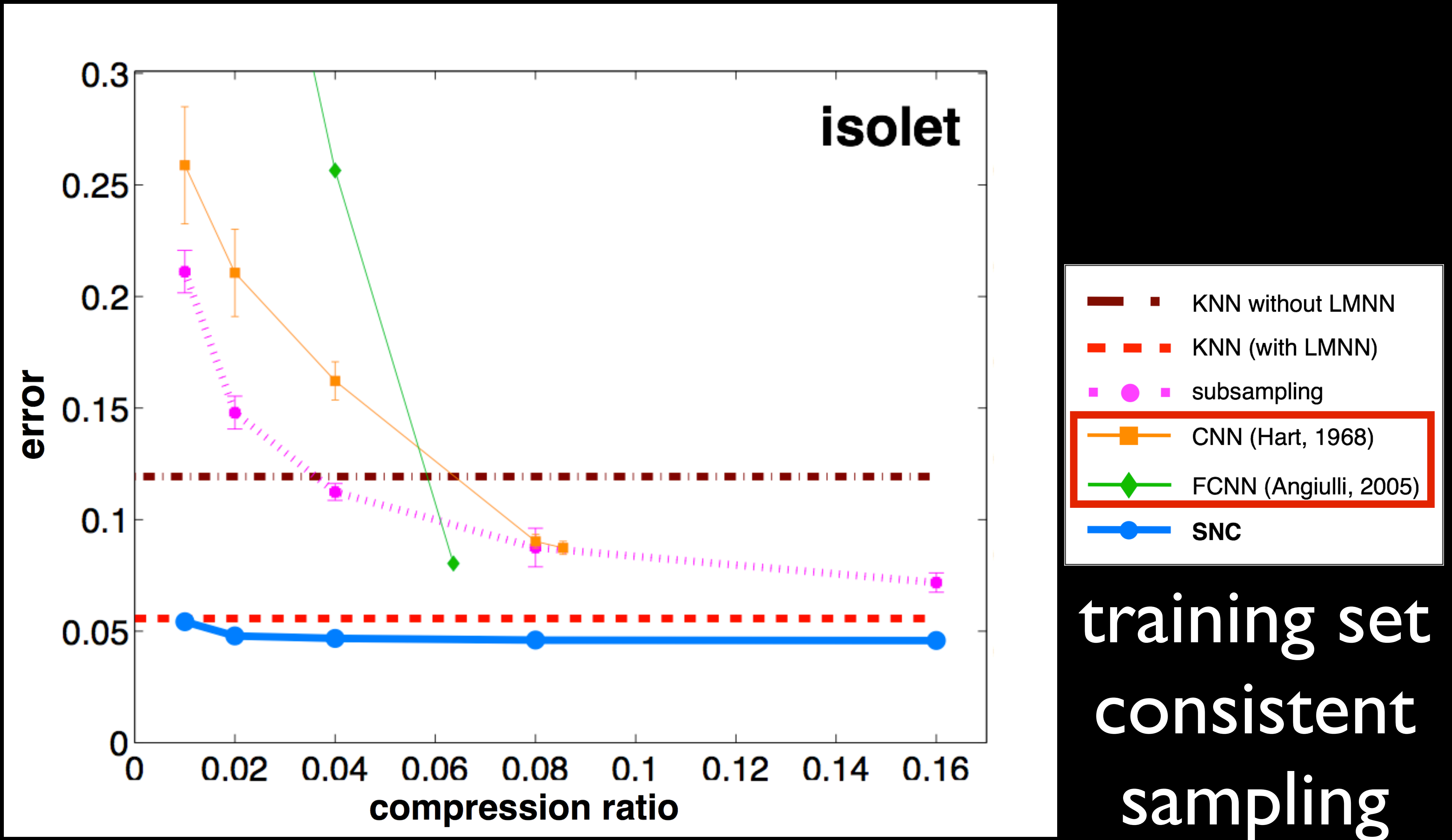
on full training set

Results



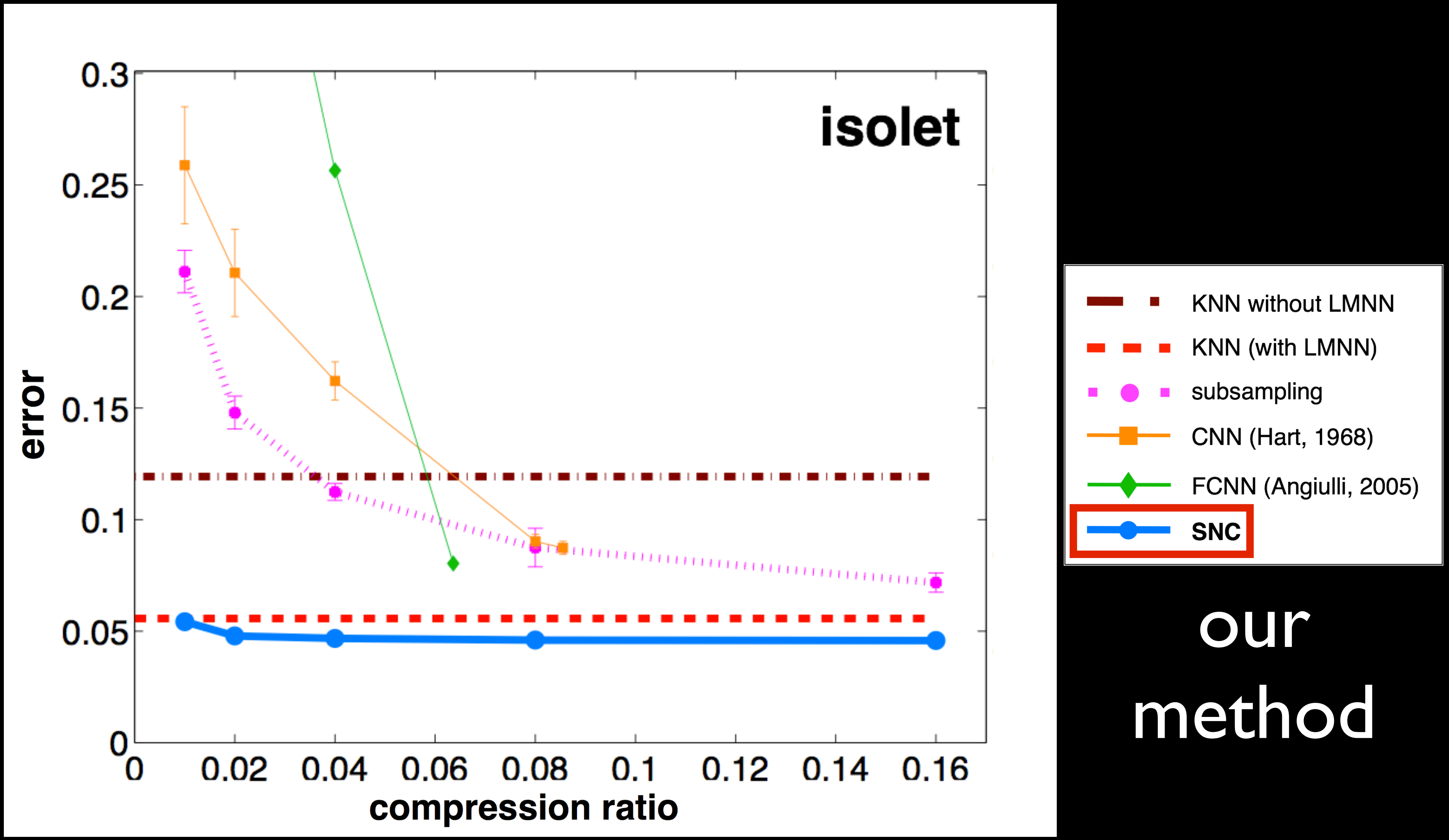
initialization

Results



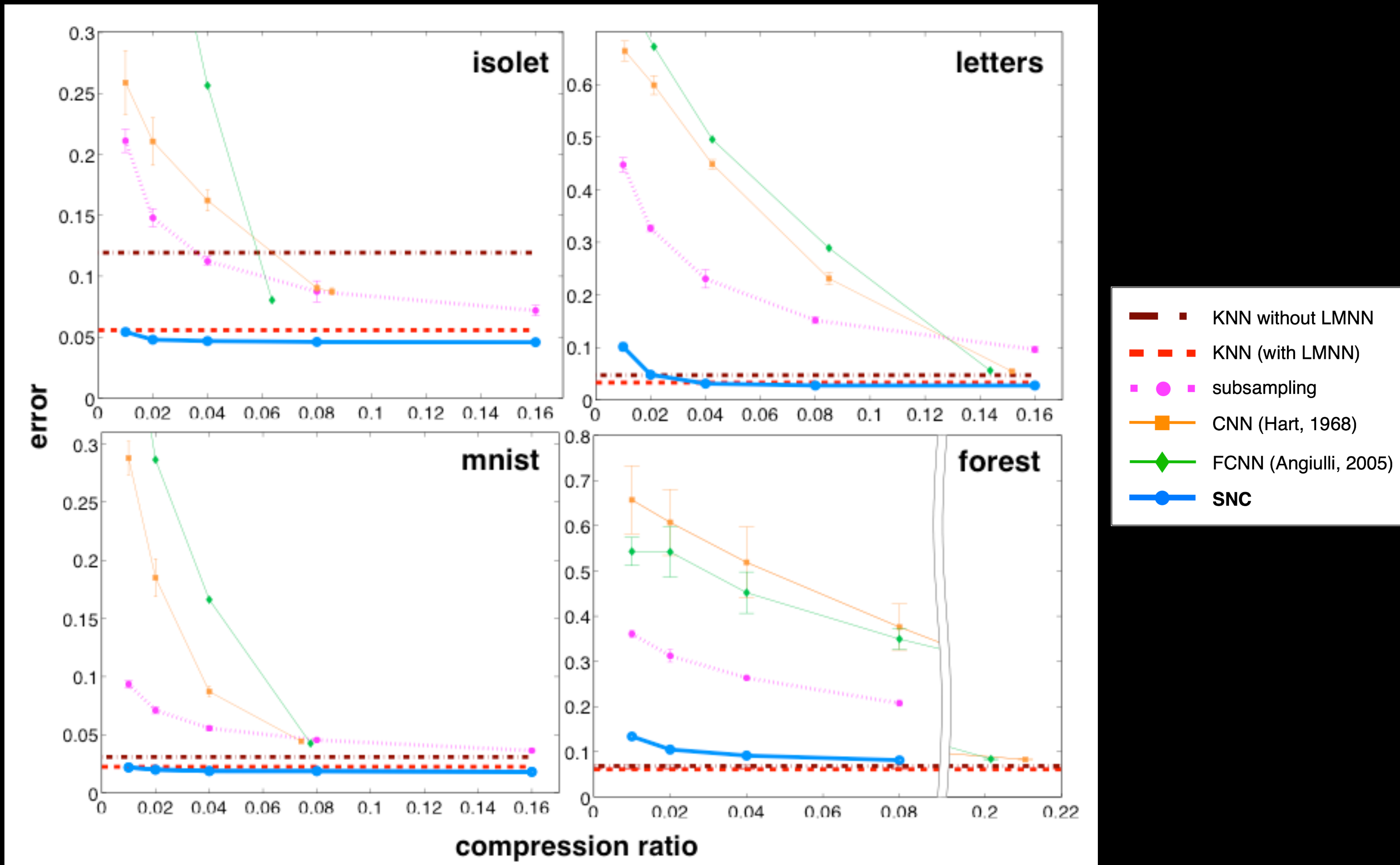
training set consistent sampling

Results



our method

Results



isolet
letters
mnist
forest
error
compression ratio
KNN without LMNN
KNN (with LMNN)
subsampling
CNN (Hart, 1968)
FCNN (Angiulli, 2005)
SNC

Results

[test-time speed-up]

Results

[test-time speed-up]

<u>complementary approaches</u>

ball-trees

locality-sensitive hashing (LSH)

Results

[test-time speed-up]

<u>complementary approaches</u>

ball-trees

locality-sensitive hashing (LSH)

DATASET	1%		
YALE-FACES	—	—	—
ISOLET	**76**	23	13
LETTERS	143	9.3	100
ADULT	**156**	56	**3.5**
W8A	146	68	39
MNIST	**136**	54	**84**
FOREST	146	3.1	12

Results

[test-time speed-up]

complementary approaches

ball-trees

locality-sensitive hashing (LSH)

compression rate

DATASET	1%		
YALE-FACES	–	–	–
ISOLET	**76**	**23**	**13**
LETTERS	143	9.3	100
ADULT	**156**	**56**	**3.5**
W8A	146	68	39
MNIST	**136**	**54**	**84**
FOREST	146	3.1	12

$$\frac{\text{1-NN full dataset}}{\text{1-NN after SNC}}$$

Results

[test-time speed-up]

complementary approaches

ball-trees

locality-sensitive hashing (LSH)

compression rate

DATASET	1%		
YALE-FACES	—	—	—
ISOLET	**76**	23	13
LETTERS	143	9.3	100
ADULT	**156**	56	**3.5**
W8A	146	68	39
MNIST	**136**	54	**84**
FOREST	146	3.1	12

$$\frac{\text{ball-trees full dataset}}{\text{ball-trees after SNC}}$$

Results

[test-time speed-up]

complementary approaches

ball-trees

locality-sensitive hashing (LSH)

compression rate

DATASET	1%		
YALE-FACES	—	—	—
ISOLET	**76**	23	13
LETTERS	143	9.3	100
ADULT	**156**	56	**3.5**
W8A	146	68	39
MNIST	**136**	54	84
FOREST	146	3.1	12

LSH full dataset

LSH after SNC

Results

[test-time speed-up]

complementary approaches

ball-trees

locality-sensitive hashing (LSH)

	SPEED-UP														
	COMPRESSION RATIO														
DATASET	1%			2%			4%			8%			16%		
YALE-FACES	–	–	–	**28**	17	3.6	**19**	11	3.5	**12**	7.3	3.2	**6.5**	4.2	2.8
ISOLET	**76**	23	13	**47**	13	13	**26**	6.8	13	14	3.7	13	**7.0**	2.0	13
LETTERS	143	9.3	100	73	6.3	61	**34**	3.6	34	**16**	2.0	17	**7.6**	1.1	8.4
ADULT	**156**	56	3.5	**75**	28	3.4	**36**	15	3.3	**17**	7.3	3.1	**7.8**	3.8	3.0
W8A	146	68	39	71	36	35	33	19	26	15	10	18	7.3	5.5	11
MNIST	**136**	54	84	**66**	29	75	**32**	16	57	**15**	8.4	37	**7.1**	3.6	17
FOREST	146	3.1	12	70	1.6	11	32	0.90	10	15	1.1	7.0	–	–	–

Results

[test-time speed-up]

<u>complementary approaches</u>

ball-trees

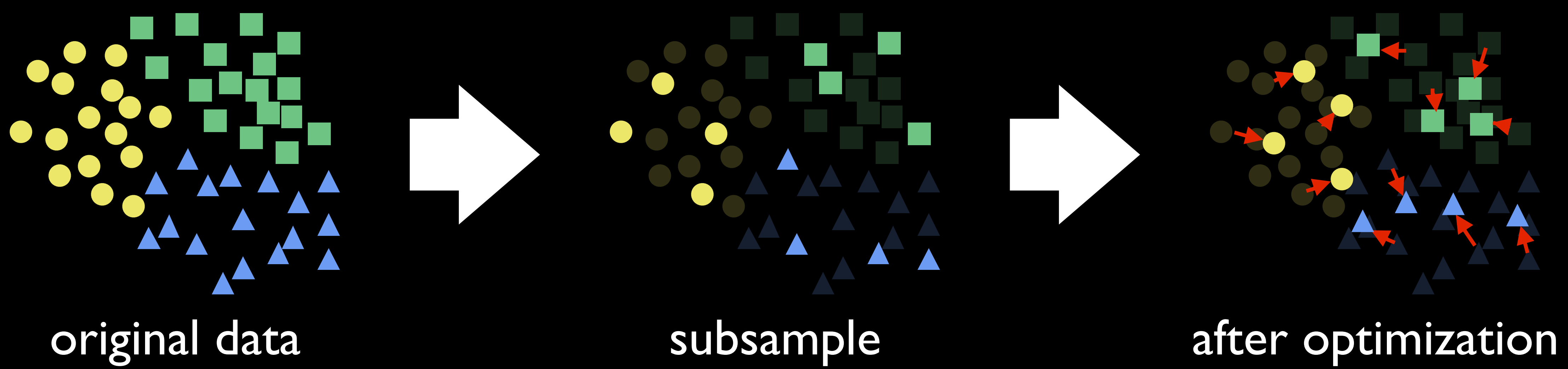
locality-sensitive hashing (LSH)

SPEED-UP															
	COMPRESSION RATIO														
DATASET	1%			2%			4%			8%			16%		
YALE-FACES	–	–	–	**28**	17	**3.6**	19	11	**3.5**	**12**	7.3	**3.2**	**6.5**	4.2	**2.8**
ISOLET	**76**	23	13	**47**	13	13	**26**	6.8	13	**14**	3.7	13	**7.0**	2.0	13
LETTERS	143	9.3	100	73	6.3	61	**34**	3.6	34	**16**	2.0	17	**7.6**	1.1	8.4
ADULT	**156**	56	**3.5**	**75**	28	**3.4**	**36**	15	**3.3**	**17**	7.3	**3.1**	**7.8**	3.8	**3.0**
W8A	146	68	39	71	36	35	33	19	26	15	10	18	7.3	5.5	11
MNIST	**136**	54	**84**	**66**	29	**75**	**32**	16	**57**	**15**	8.4	**37**	**7.1**	3.6	**17**
FOREST	146	3.1	12	70	1.6	11	32	0.90	10	15	1.1	7.0	–	–	–

Summary

Stochastic Neighbor Compression

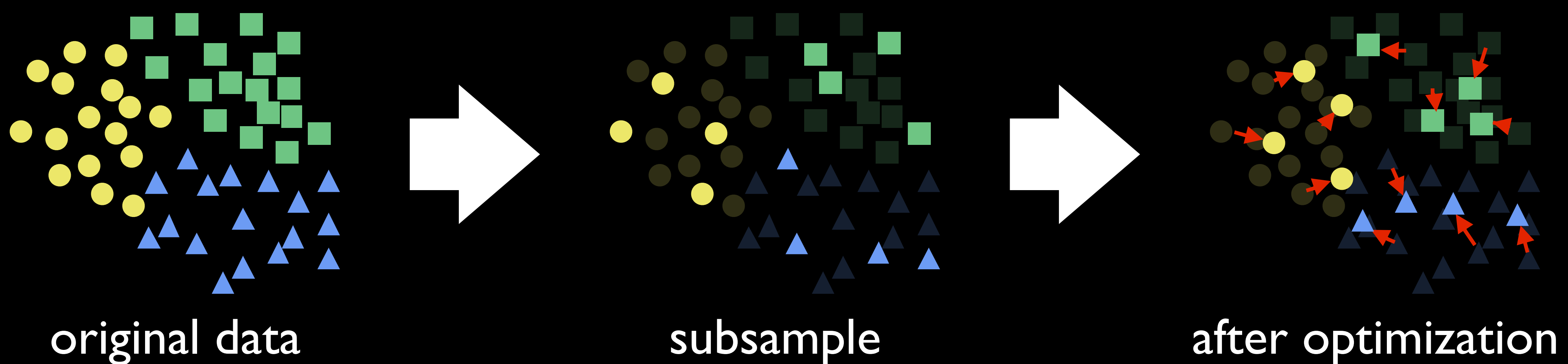
- learns a compressed training set for NN classifier



Summary

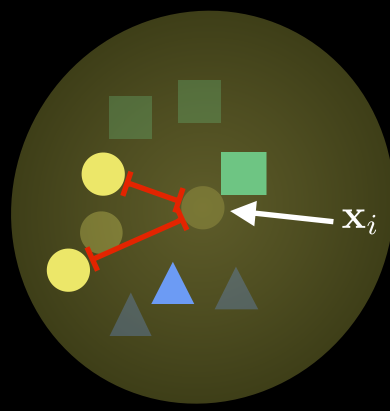
Stochastic Neighbor Compression

- learns a compressed training set for NN classifier



- Stochastic Neighborhood

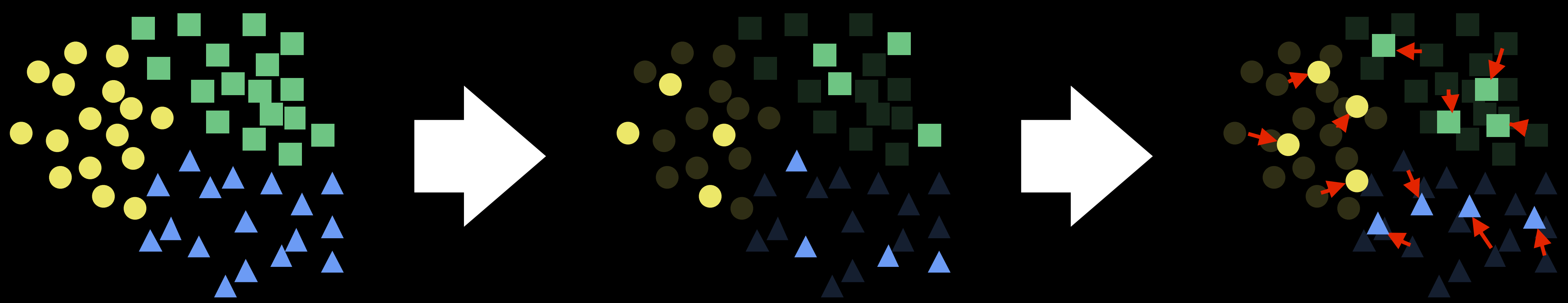
[Hinton & Roweis, 2002]



Summary

Stochastic Neighbor Compression

- learns a compressed training set for NN classifier

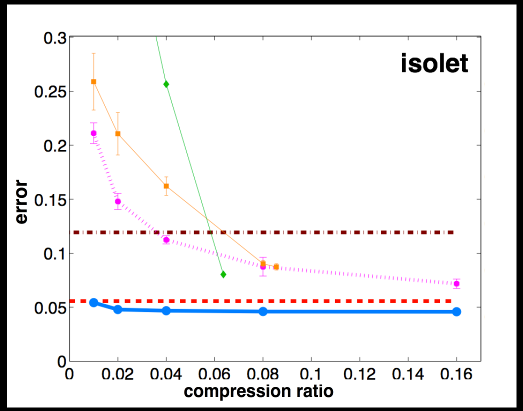


original data subsample after optimization

- Stochastic Neighborhood

  [Hinton & Roweis, 2002]

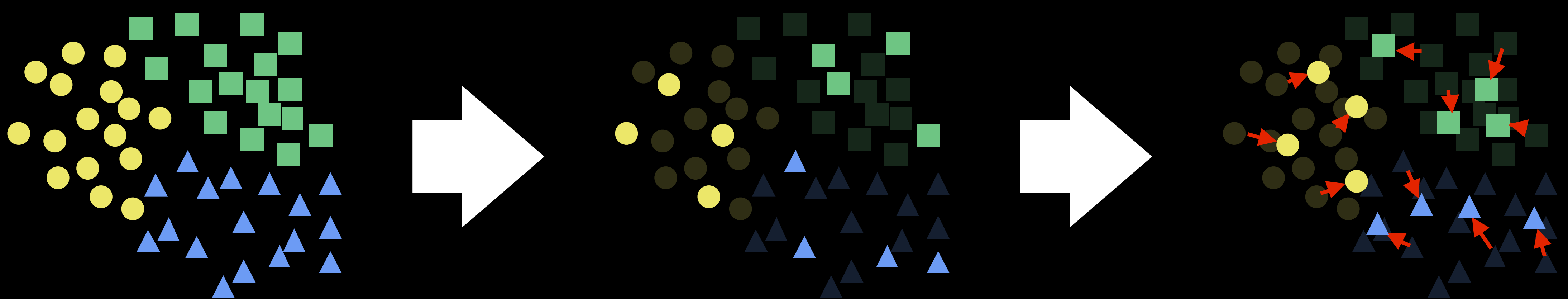
- Compression by 96% without error increase in 5/7 cases



Summary

Stochastic Neighbor Compression

- learns a compressed training set for NN classifier



original data | subsample | after optimization

- 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

	Speed-up														
	Compression Ratio														
Dataset	1%			2%			4%			8%			16%		
YALE-FACES	–	–	–	**28**	17	3.6	**19**	11	3.5	**12**	7.3	3.2	**6.5**	4.2	2.8
ISOLET	**76**	23	13	**47**	13	13	**26**	6.8	13	**14**	3.7	13	**7.0**	2.0	13
LETTERS	143	9.3	100	73	6.3	61	**34**	3.6	34	**16**	2.0	17	**7.6**	1.1	8.4
ADULT	**156**	56	3.5	**75**	28	3.4	**36**	15	3.3	**17**	7.3	3.1	**7.8**	3.8	3.0
W8A	146	68	39	71	36	35	33	19	26	15	10	18	7.3	5.5	11
MNIST	**136**	54	84	**66**	29	75	**32**	16	57	**15**	8.4	37	**7.1**	3.6	17
FOREST	146	3.1	12	70	1.6	11	32	0.90	10	15	1.1	7.0	–	–	–

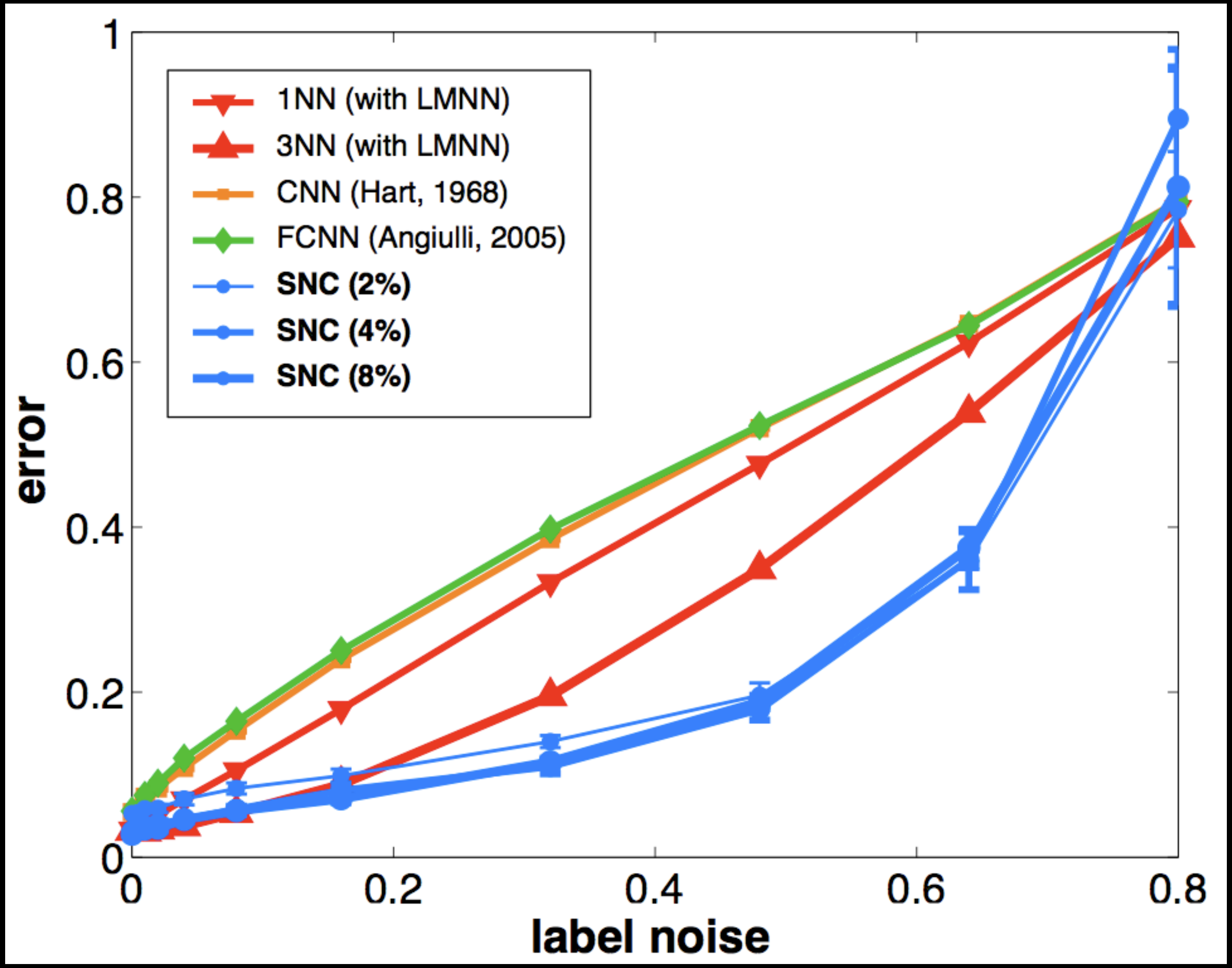
Thank you!
Questions?

Results

[robustness to label noise]

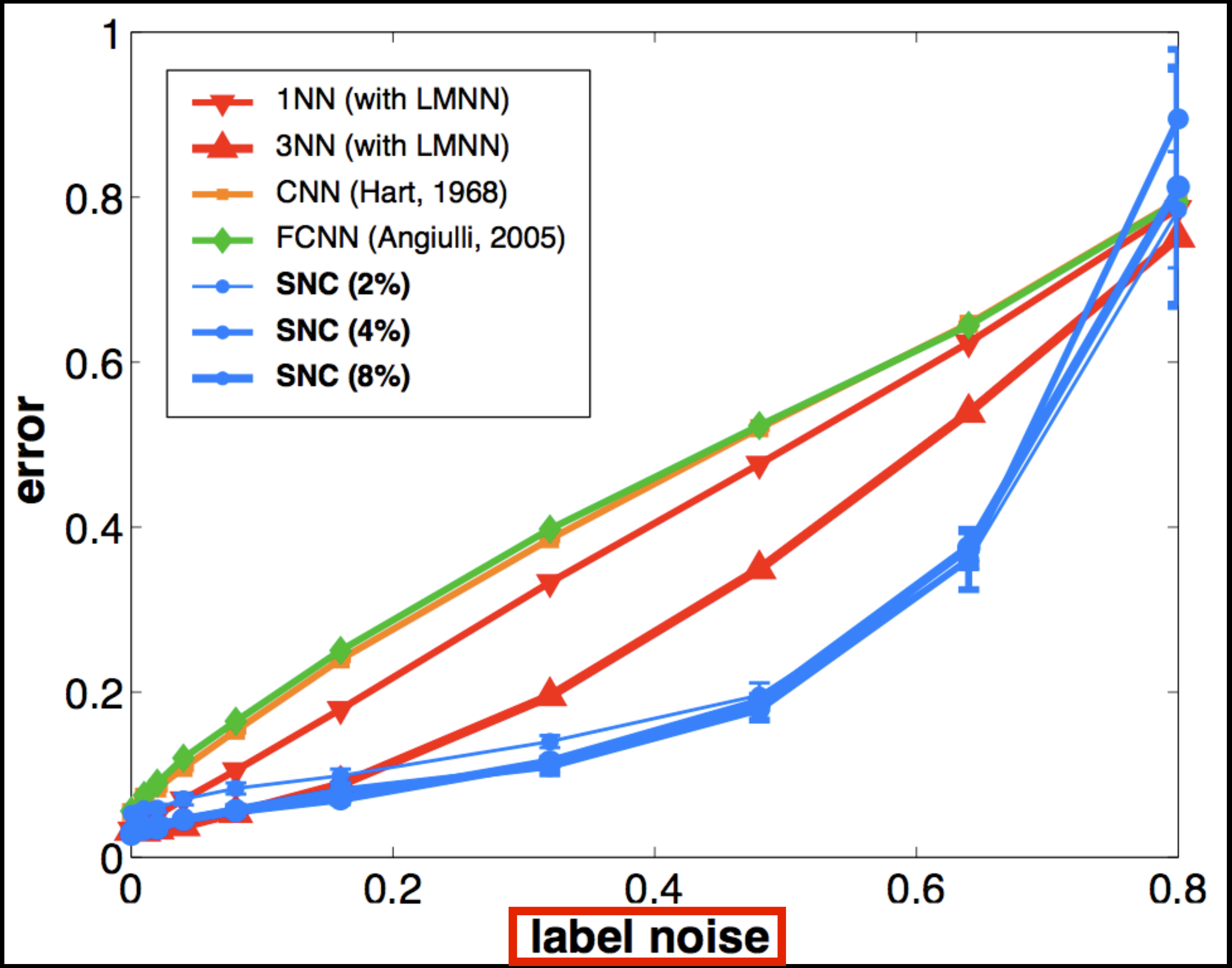
Results

[robustness to label noise]



Results

[robustness to label noise]

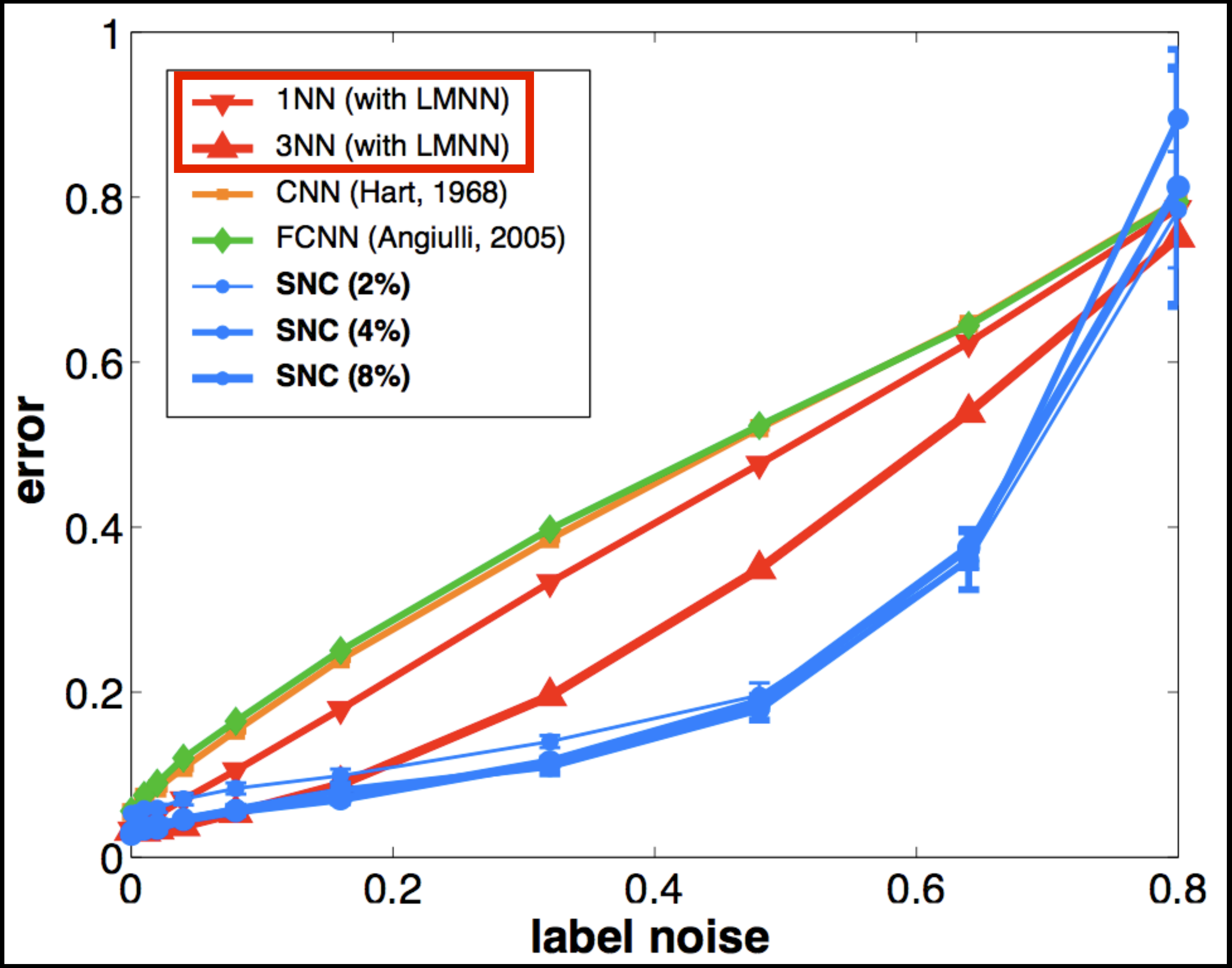


noise added to training labels

Results

[robustness to label noise]

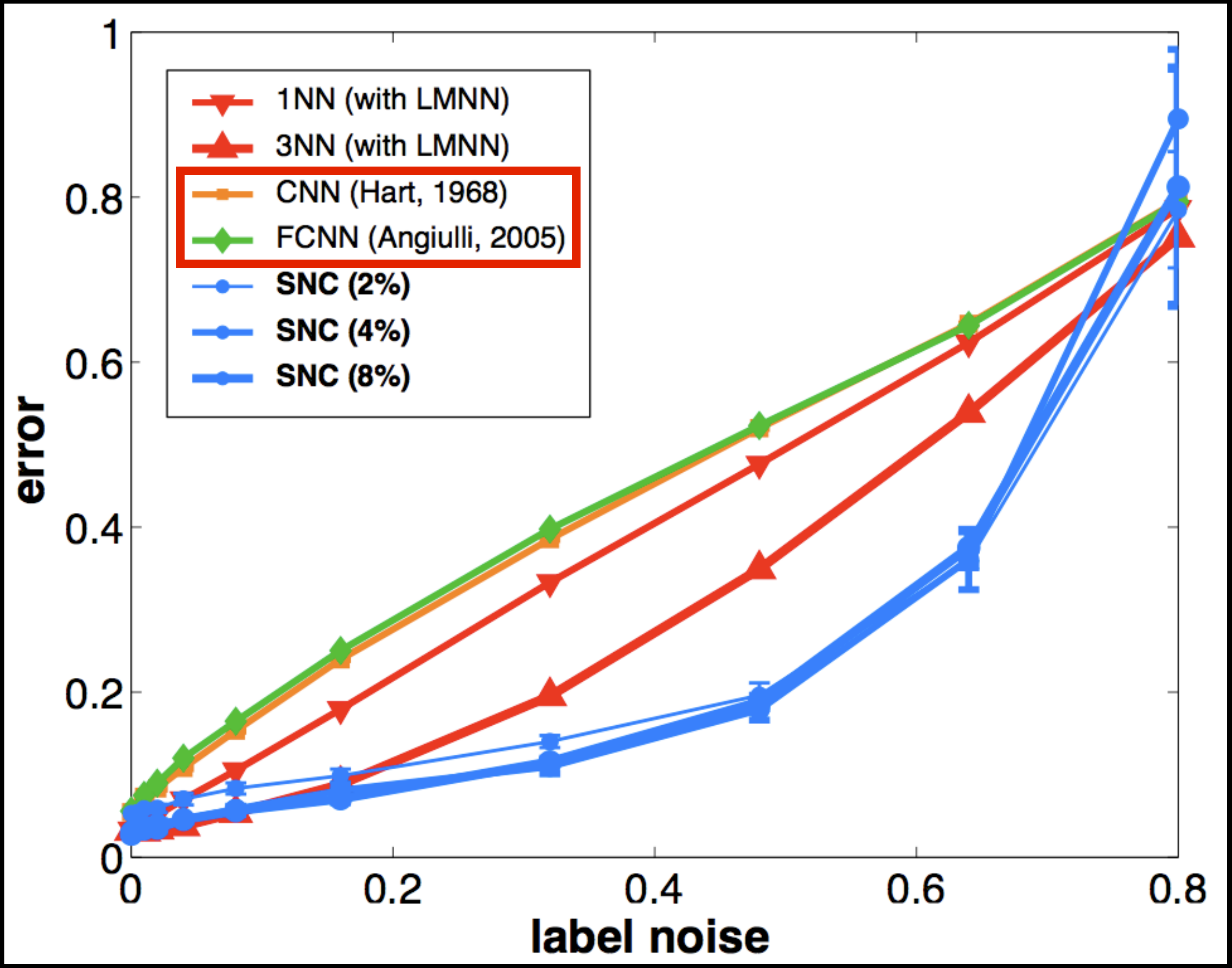
using larger k



Results

[robustness to label noise]

training set consistent sampling



Results

[robustness to label noise]

our method