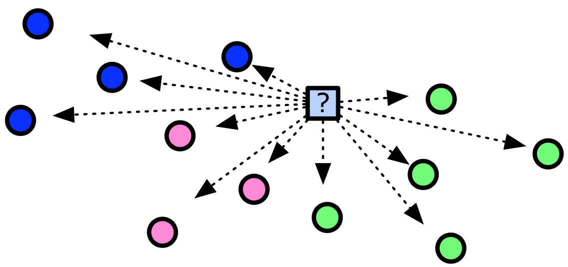
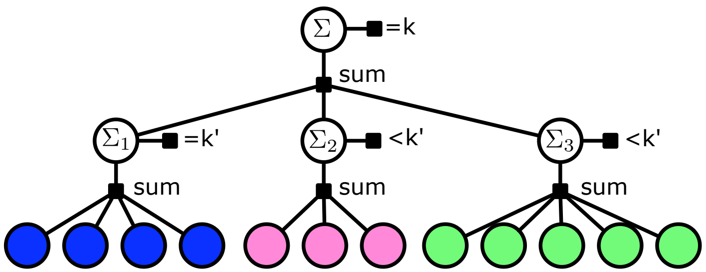
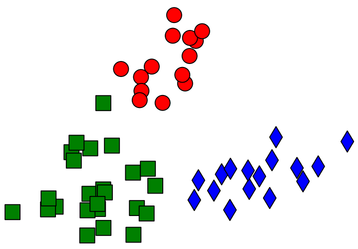
Stochastic k-Neighborhood Selection for Supervised and Unsupervised Learning







Danny Tarlow

Kevin Swersky

Ilya Sutskever

Laurent Charlin

Richard Zemel

University of Toronto Machine Learning Seminar

Feb 21, 2013

Distance Metric Learning

Distance metrics are everywhere.

But they're arbitrary! Dimensions are scaled weirdly, and even if they're normalized, it's not clear that Euclidean distance means much.

So learning sounds nice, but what you learn should depend on the task.

A really common task is kNN. Let's look at how to learn distance metrics for that.

Popular Approaches for Distance Metric Learning

Some satisfying properties

- Based on local structure (doesn't have to pull all points into one region)

Some unsatisfying properties

- Initial choice of target neighbors is difficult
- Choice of objective function has reasonable forces (pushes and pulls), but beyond that, it is pretty heuristic.
- No probabilistic interpretation.

"Tar
mus
ahea

Large margin nearest neighbors (LMNN)

[Weinberger et al., NIPS 2006]

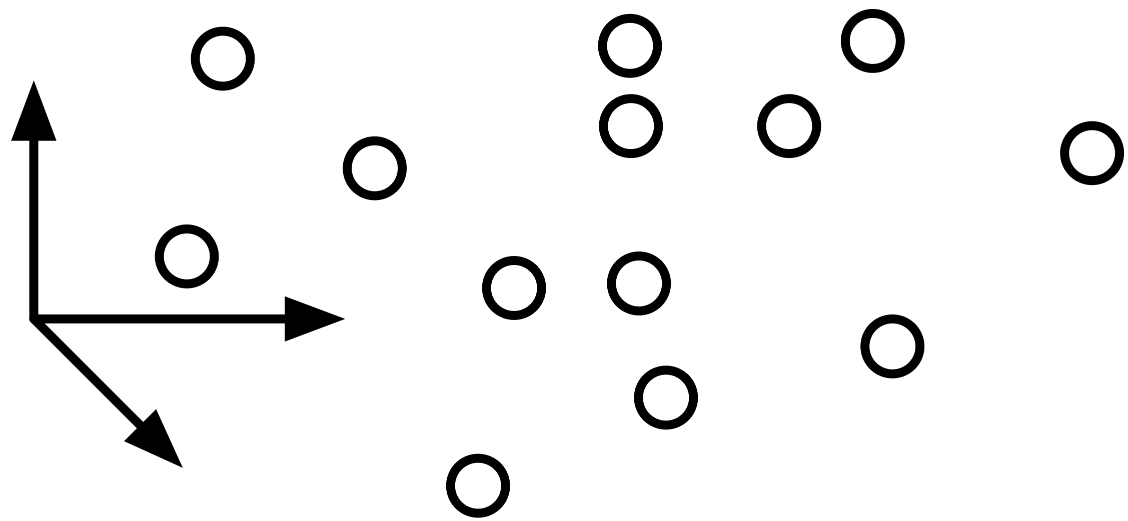
Probabilistic Formulations for Distance Metric Learning

Our goal: give a probabilistic interpretation of kNN and properly learn a model based upon this interpretation.

Related work that kind of does this: Neighborhood Components Analysis (NCA). Our approach is a direct generalization.

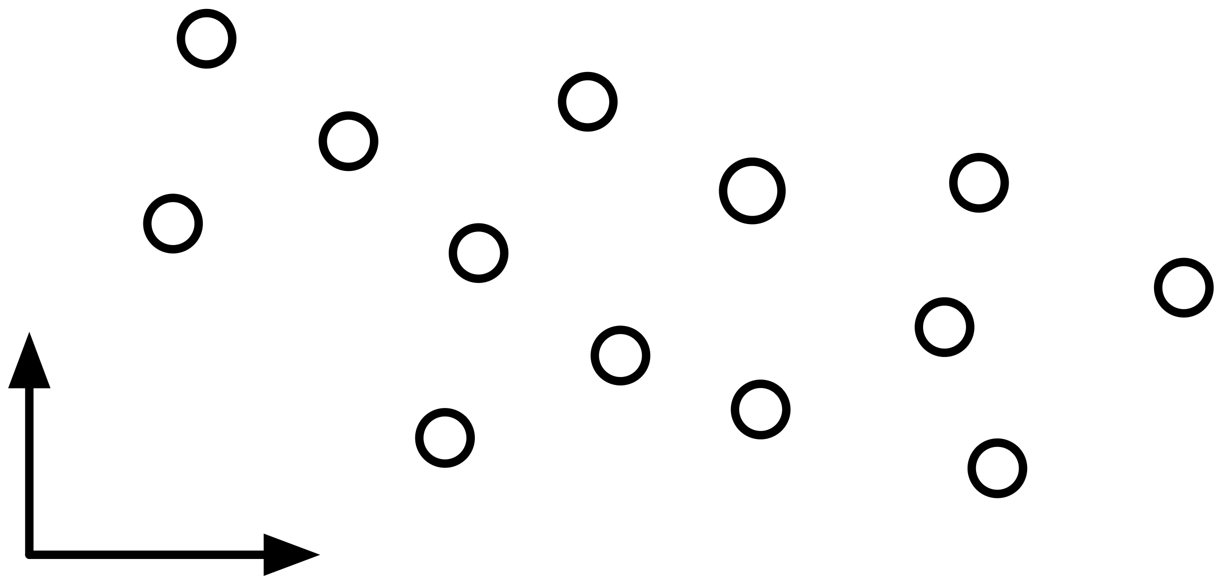
Generative Model

1. Given points $X \in \mathbb{R}^{N \times D}$



Generative Model

2. Project into P dimensional space via $A \in \mathbb{R}^{D \times P}$



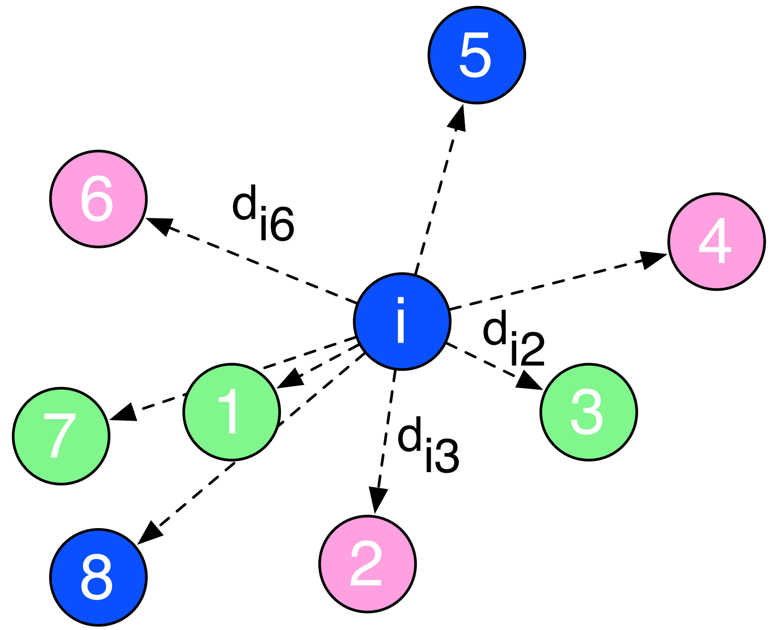
$$E \in \mathbb{R}^{N \times P} = XA$$

$$d_{ij} = ||e_i - e_j||^2 = ||Ax_i - Ax_j||^2 = (x_i - x_j)^T A^T A(x_i - x_j)$$

Neighborhood Component Analysis (NCA)

[Goldberger et al., 2004]

Given a query point i.
We select neighbors
randomly according to d.



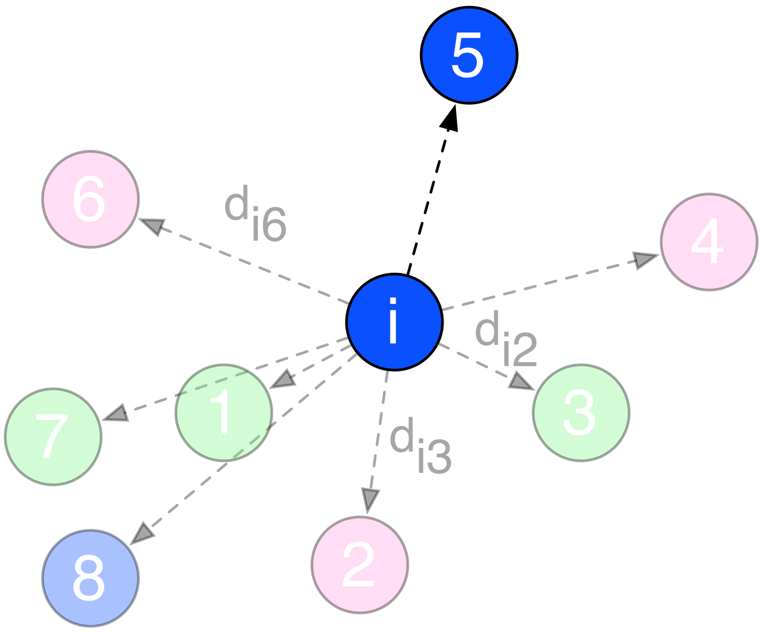
Question: what is the
probability that a randomly
selected neighbor will belong
to the correct (blue) class?

Neighborhood Component Analysis (NCA)

[Goldberger et al., 2004]

Question: what is the probability that a randomly selected neighbor will belong to the correct (blue) class?

$$P_{i5} = \frac{\exp(-d_{i5})}{Z_{0i}}$$

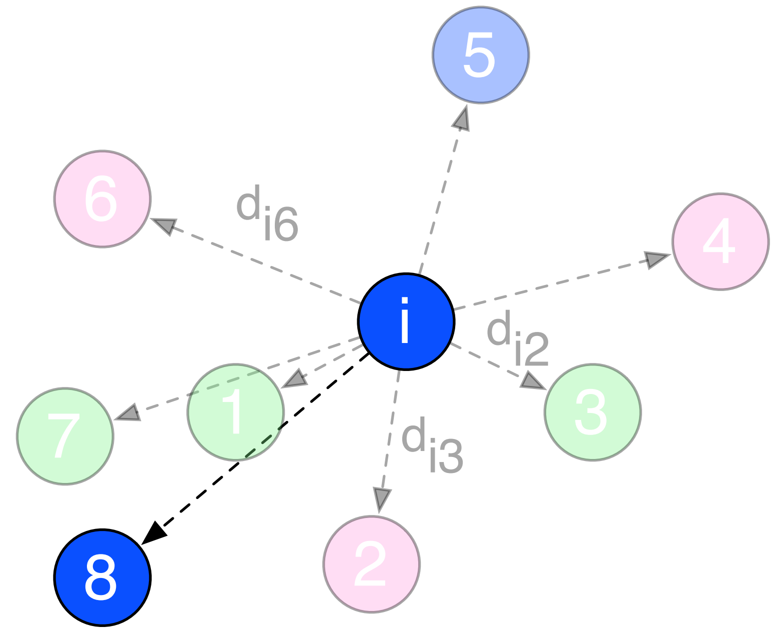


$$Z_{0i} = \sum_{j \neq i} \exp(-d_{ij})$$

Neighborhood Component Analysis (NCA)

[Goldberger et al., 2004]

Question: what is the probability that a randomly selected neighbor will belong to the correct (blue) class?



$$P_{i5} = \frac{\exp(-d_{i5})}{Z_{0i}}$$

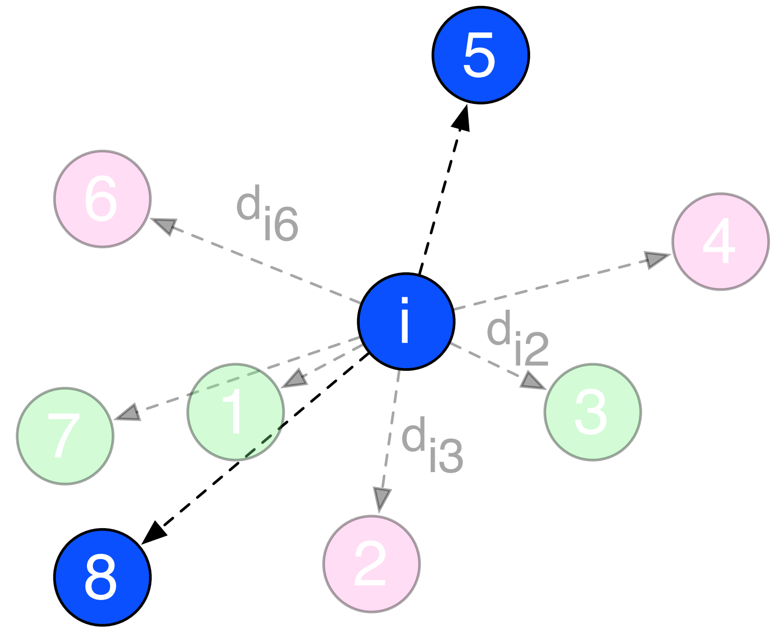
$$P_{i8} = \frac{\exp(-d_{i8})}{Z_{0i}}$$

$$Z_{0i} = \sum_{j \neq i} \exp(-d_{ij})$$

Neighborhood Component Analysis (NCA)

[Goldberger et al., 2004]

Question: what is the probability that a randomly selected neighbor will belong to the correct (blue) class?



$$P_{i5} = \frac{\exp(-d_{i5})}{Z_{0i}}$$

$$P_{i8} = \frac{\exp(-d_{i8})}{Z_{0i}}$$

$$P_i = P_{i5} + P_{i8}$$

$$Z_{0i} = \sum_{j \neq i} \exp(-d_{ij})$$

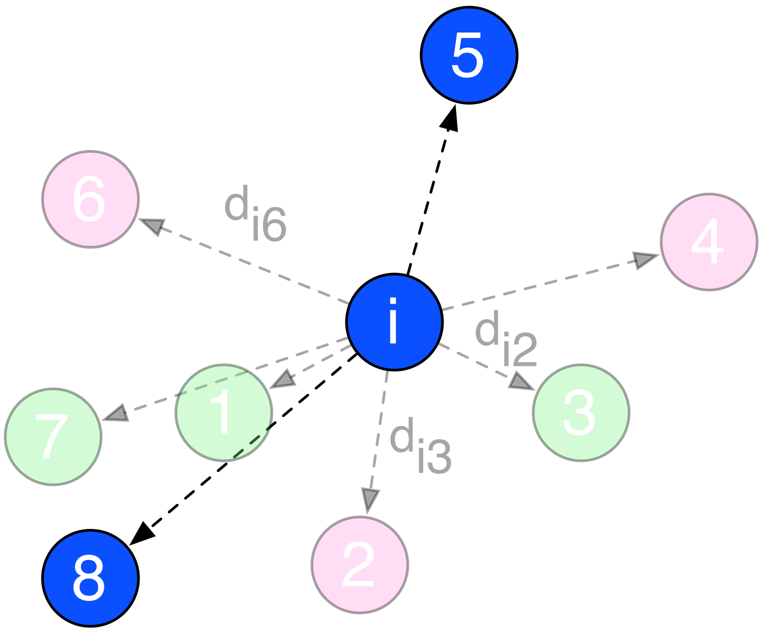
Neighborhood Component Analysis (NCA)

[Goldberger et al., 2004]

Another way to write this:

$$P_i = \sum_{j \neq i} P_{ij} [y_i = y_j]$$

$$= \frac{Z_{1i}}{Z_{0i}}$$



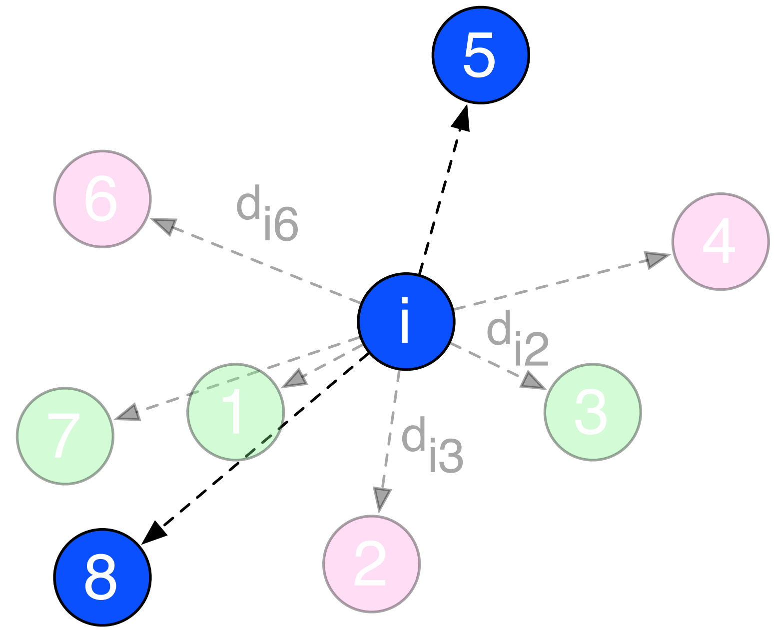
$$Z_{1i} = \exp(-d_{i5}) + \exp(-d_{i8})$$

(y are the class labels)

Neighborhood Component Analysis (NCA)

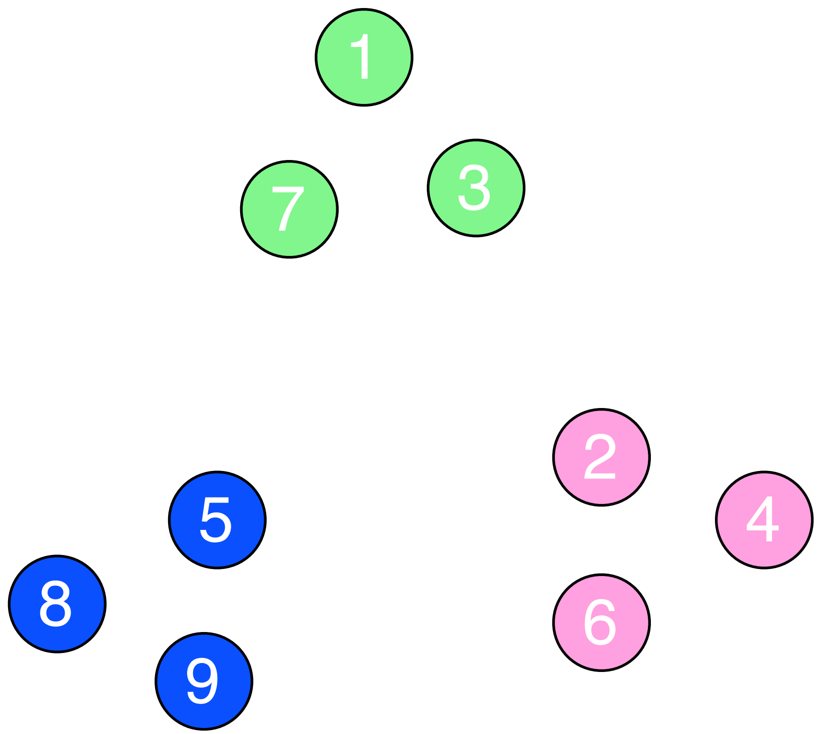
[Goldberger et al., 2004]

Objective: maximize the log-likelihood of stochastically selecting neighbors of the same class.



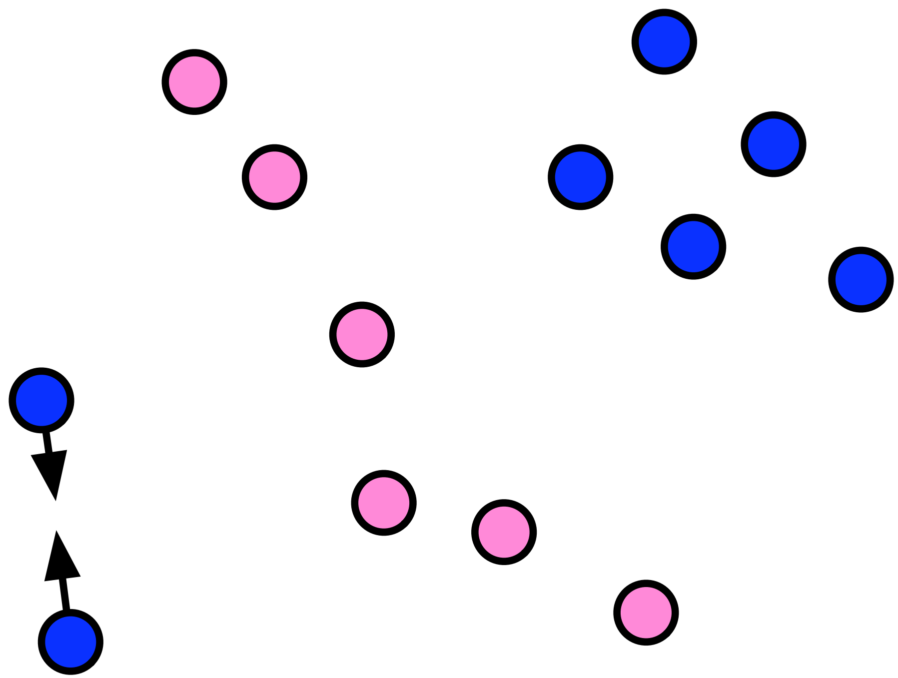
$$\mathcal{L} = \sum_i \log\Big(\sum_{j \neq i} P_{ij}[y_i = y_j]\Big)$$

After Learning



We might hope to learn a projection that looks like this.

Problem with 1-NCA



NCA is happy if points pair up and ignore global structure. This is not ideal if we want k > 1.

k-Neighborhood Component Analysis (k-NCA)

NCA: $$P_i = \sum_{j \neq i} P_{ij} [y_i = y_j]$$

[] is the identity function

k-NCA: $$P_i = \sum_{s \in S : |s| = k} P_i(s) [Maj(y_s) = y_i]$$

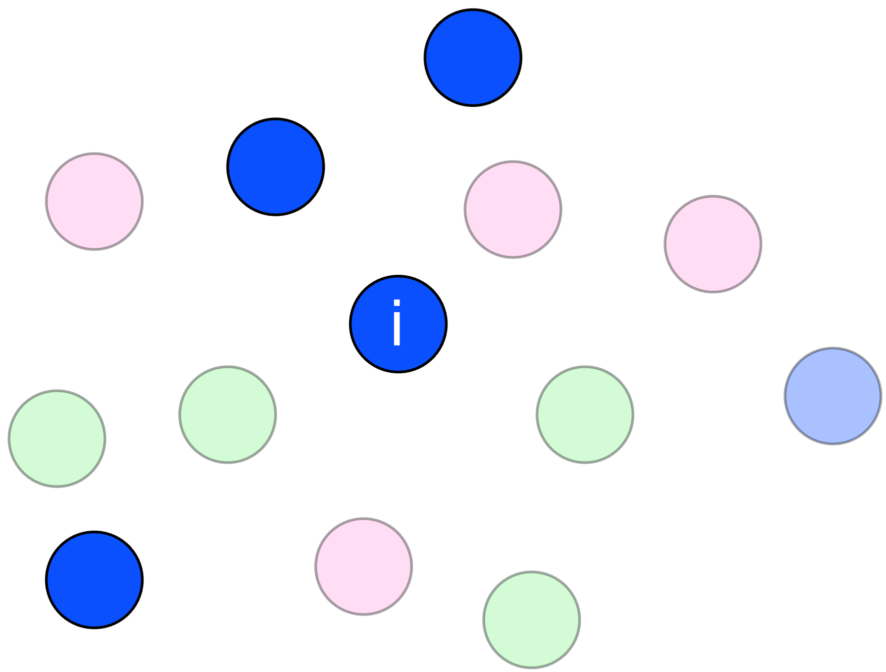
(S is all sets of k neighbors of point i)

$$= \frac{\sum_{s \in S : |s| = k} \exp(-\sum_{j \in s} d_{ij}) [Maj(y_s) = y_i]}{\sum_{s \in S : |s| = k} \exp(-\sum_{j \in s} d_{ij})}$$

Setting k=1 recovers NCA.

k-Neighborhood Component Analysis (k-NCA)

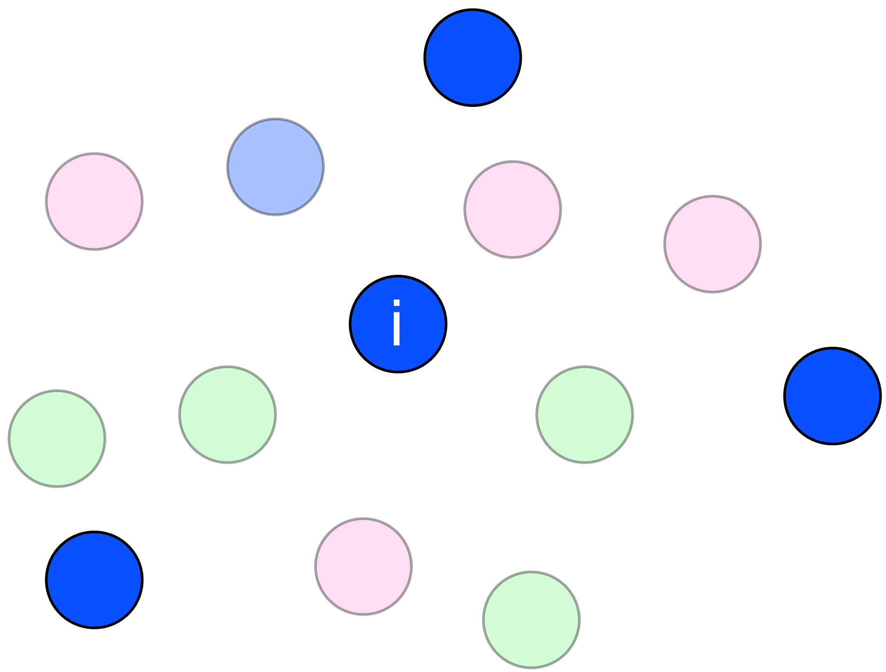
Computing the numerator of the distribution P_i



Stochastically choose k neighbors such that the majority is blue.

k-Neighborhood Component Analysis (k-NCA)

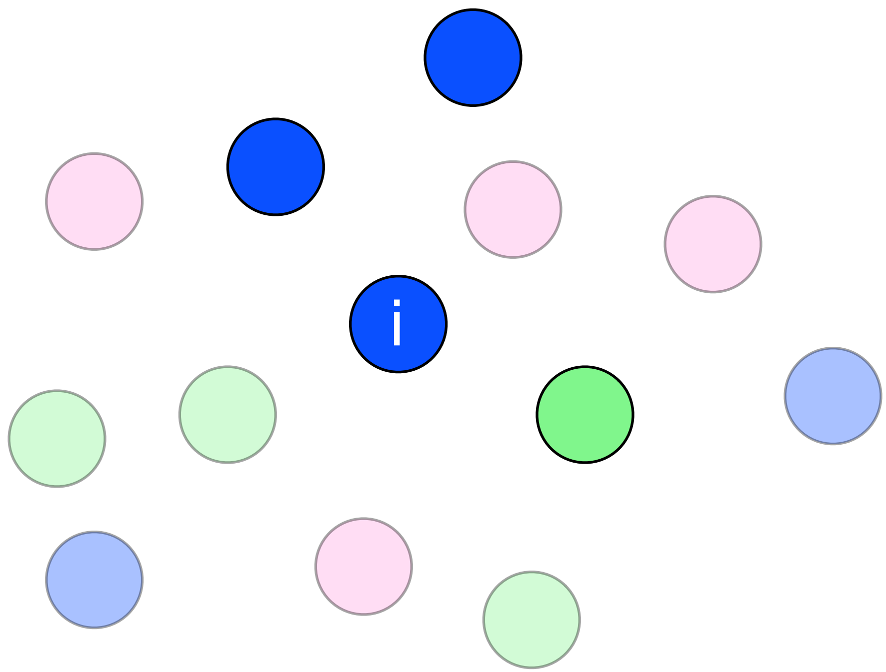
Computing the numerator of the distribution P_i



Stochastically choose k neighbors such that the majority is blue.

k-Neighborhood Component Analysis (k-NCA)

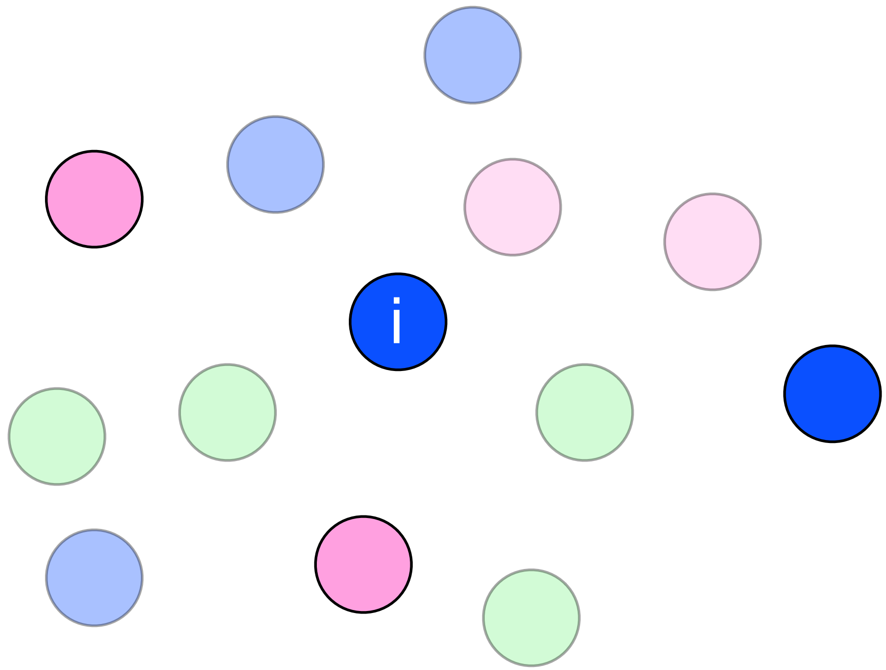
Computing the numerator of the distribution P_i



Stochastically choose k neighbors such that the majority is blue.

k-Neighborhood Component Analysis (k-NCA)

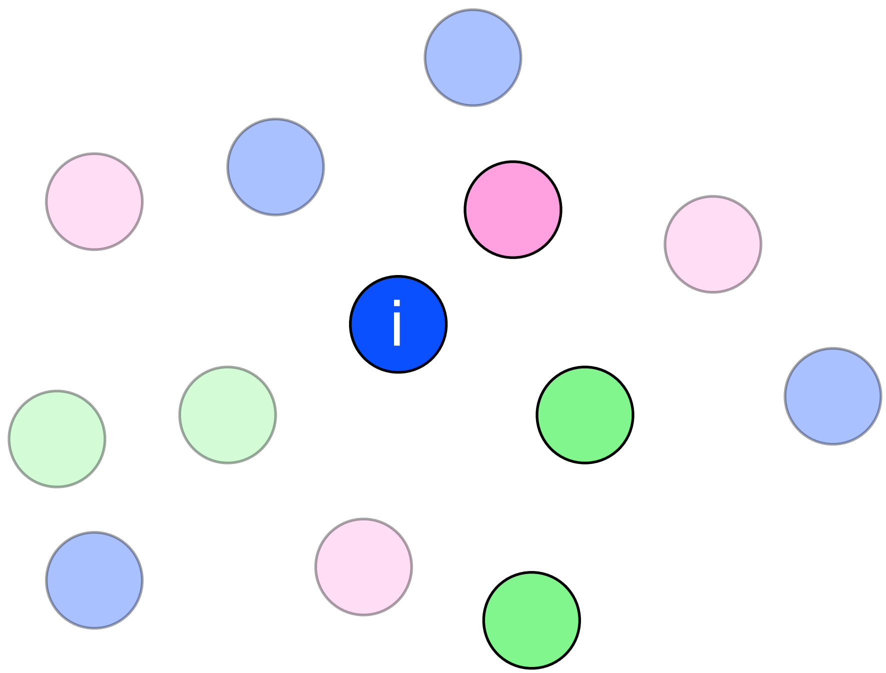
Computing the denominator of the distribution P_i



Stochastically choose subsets of k neighbors.

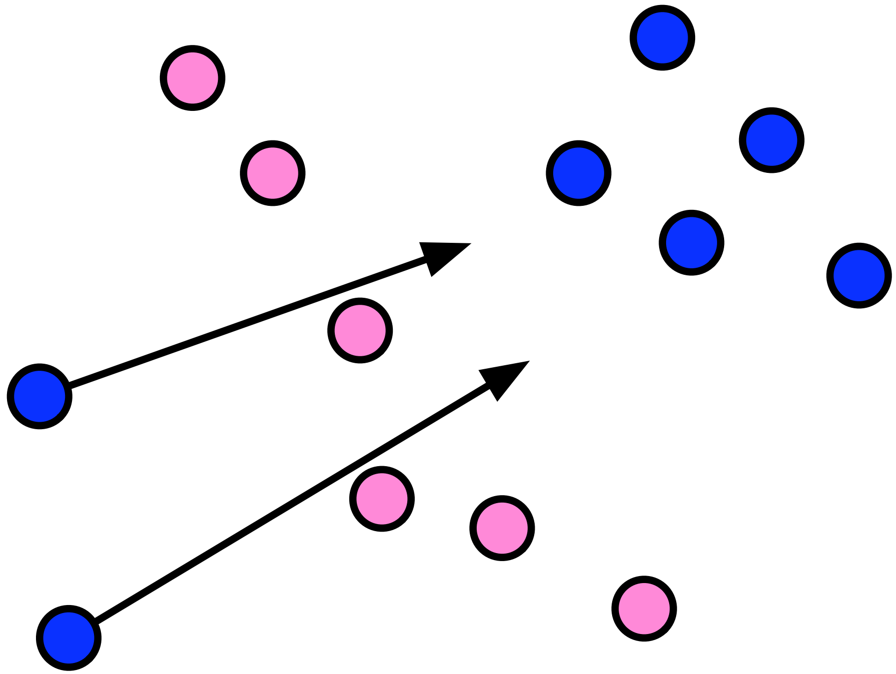
k-Neighborhood Component Analysis (k-NCA)

Computing the denominator of the distribution P_i



Stochastically choose subsets of k neighbors.

k-NCA Intuition



k-NCA puts more pressure on points to form bigger clusters.

k-NCA Objective

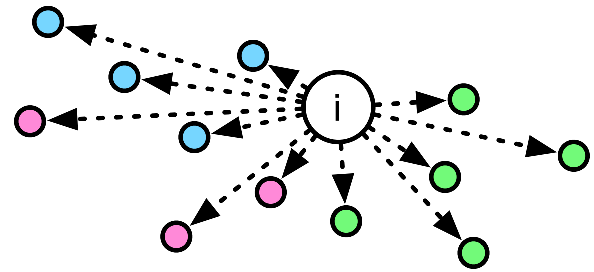
Given: inputs X, labels y, neighborhood size k.

$$\mathcal{L}(\mathcal{A}) = \sum_i \log(\sum_{s \in S_i} P_i(s|A;k)[Maj(y_s) = y_i])$$

Learning: find A that (locally) maximizes this.

Technical challenge: efficiently compute $\mathcal{L}(A)$ and $\frac{\partial \mathcal{L}}{\partial A}$

Factor Graph Formulation

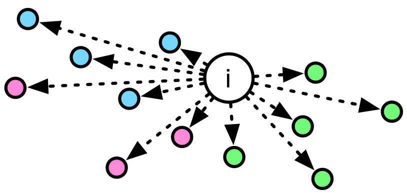


$$\mathcal{L}(A;k) = \prod_i \underbrace{\sum_{s \in S_i} P_i(s|A;k) \cdot 1\{\textsc{Majority}(\boldsymbol{y}_s) = y_i\}}_{\text{Focus on a single } i}$$

$$\frac{\sum_{s:|s|=k} \exp\{-\sum_{j \in s} d_{ij}(A)\} \cdot 1\{\textsc{Maj}(\boldsymbol{y}_s) = y_i\}}{\sum_{s:|s|=k} \exp\{-\sum_{j \in s} d_{ij}(A)\}}$$

Factor Graph Formulation

$$\frac{\sum_{s:|s|=k} \exp\{-\sum_{j\in s} d_{ij}(A)\} \cdot 1\{\textsc{Maj}(\boldsymbol{y}_s) = y_i\}}{\sum_{s:|s|=k} \exp\{-\sum_{j\in s} d_{ij}(A)\}}$$



Step 1: Split Majority function into cases.

Switch(k'=|$\mathbf{y}_s$ = y_i|) // # neighbors w/ label y_i

Maj(y_s) = 1 iff forall c != y_i, |$\mathbf{y}_s$=c| < k'

Step 2: Constrain total # neighbors chosen to be k.

$$\frac{\sum_{s:|s|=k} \exp\{-\sum_{j\in s} d_{ij}(A)\} \cdot 1\{\text{MAJ}(\boldsymbol{y}_s) = y_i\}}{\sum_{s:|s|=k} \exp\{-\sum_{j\in s} d_{ij}(A)\}}$$

Count total number of neighbors chosen

Exactly k total neighbors must be chosen

Total number of "blue" neighbors chosen

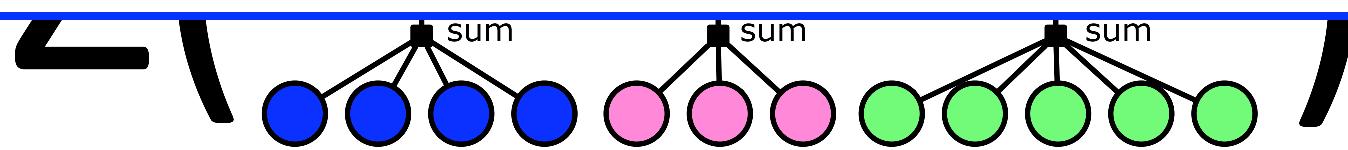
Less than k "pink" neighbors are chosen

Binary variable: is j chosen as neighbor?

chosen

At this point, everything is just a matter of inference in these factor graphs

- Partition functions: give objective
- Marginals: give gradients



Assume y_i = 'blue"

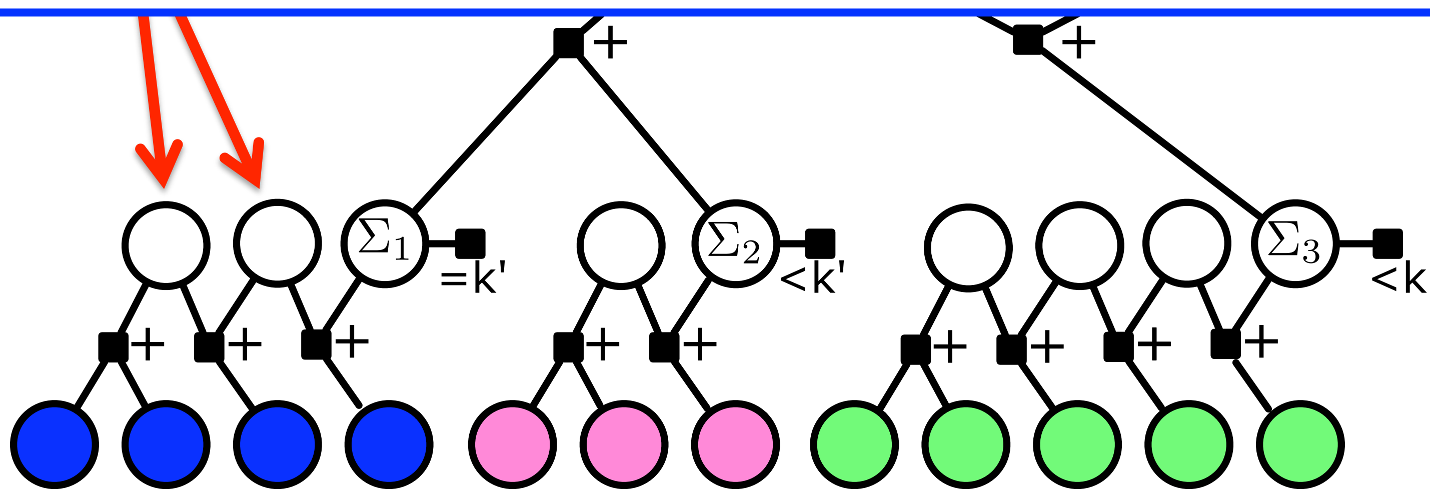
Sum-Product Inference

Lower level messages: O(k) time each

Upper level messages: O(k²) time each

Total runtime: O(Nk + C k²)*

* Although slightly better is possible asymptotically. See Tarlow et al., UAI 2012.



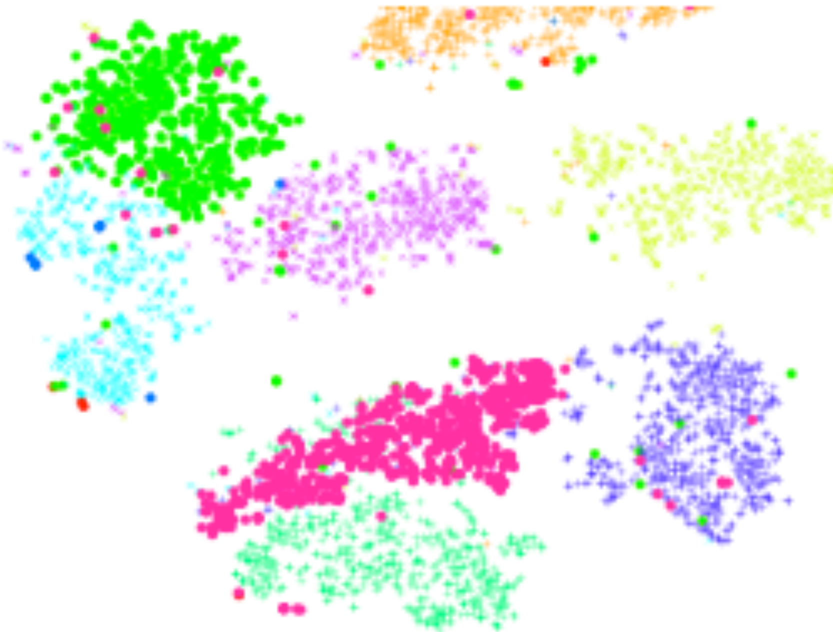
Alternative Version

- Instead of Majority($\mathbf{y}_s$)=y_i function, use All($\mathbf{y}_s$)=y_i.
  - Computation gets a little easier (just one k' needed)
  - Loses the *k*NN interpretation.
  - Exerts more pressure for homogeneity; tries to create a larger margin between classes.
  - Usually works a little better.

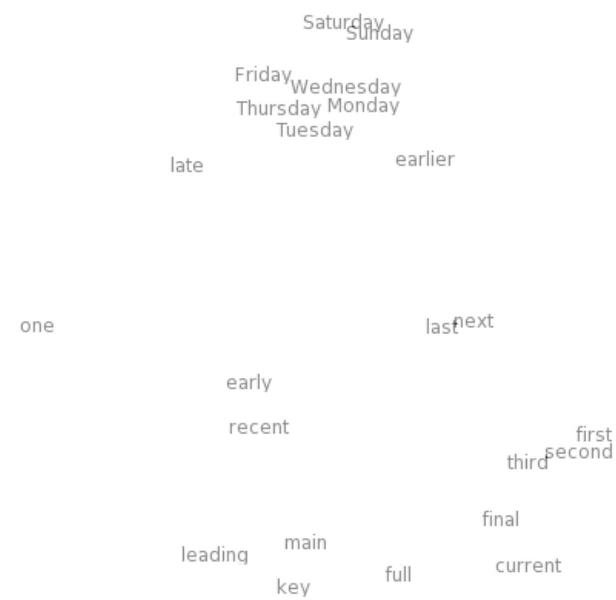
Unsupervised Learning with t-SNE

[van der Maaten and Hinton, 2008]

Visualize the structure of data in a 2D embedding.



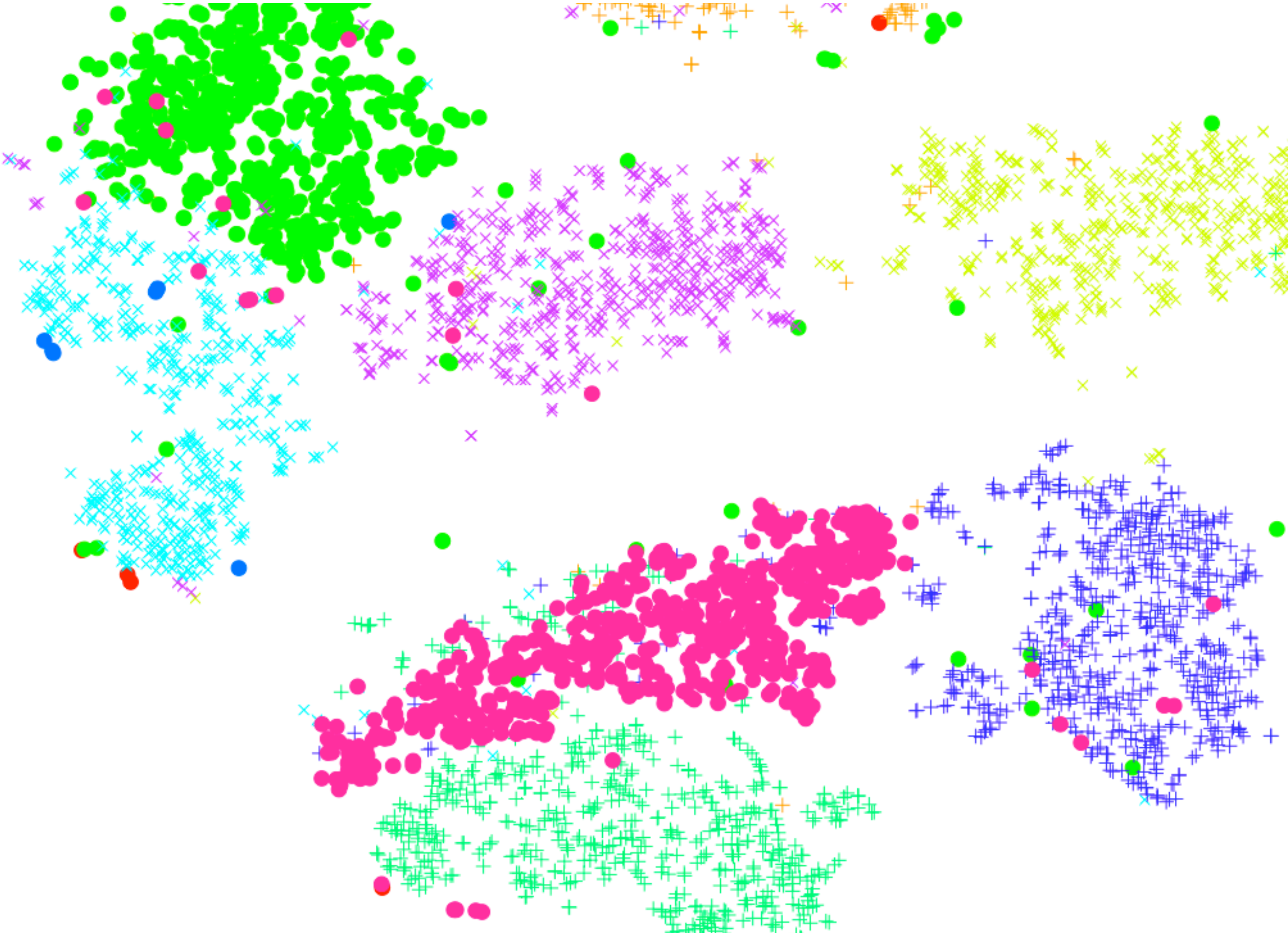
[van der Maaten & Hinton, JMLR 2008]



[Turian, http://metaoptimize.com/projects/wordreprs/]

- Each input point x maps to an embedding point e.
- SNE tries to preserve relative pairwise distances as faithfully as possible.

Problem with t-SNE (also based on k=1)



[van der Maaten & Hinton, JMLR 2008]

Unsupervised Learning with t-SNE

[van der Maaten and Hinton, 2008]

Data distribution: $P_{ij} \propto \exp(-d^p_{ij})$

Embedding distribution: $Q_{ij} \propto \exp(-d^q_{ij})$

Objective (minimize wrt e):

$$\sum_i KL(P_i||Q_i) = -\sum_i \sum_j P_{ij} \log Q_{ij} + const$$

t-SNE: $d^p_{ij} = \frac{||x_i - x_j||^2}{\sigma_i^2}$ $d^q_{ij} = (1 + ||e_i - e_j||^2)^{-1}$

kt-SNE

Data distribution: $$P_i(s) \propto \exp(-\sum_{j \in s} d_{ij}^p)$$

Embedding distribution: $$Q_i(s) \propto \exp(-\sum_{j \in s} d_{ij}^q)$$

Objective: $$\sum_i \sum_{s \in S : |s| = k} P_i(s) \log Q_i(s)$$

Minimize objective wrt e

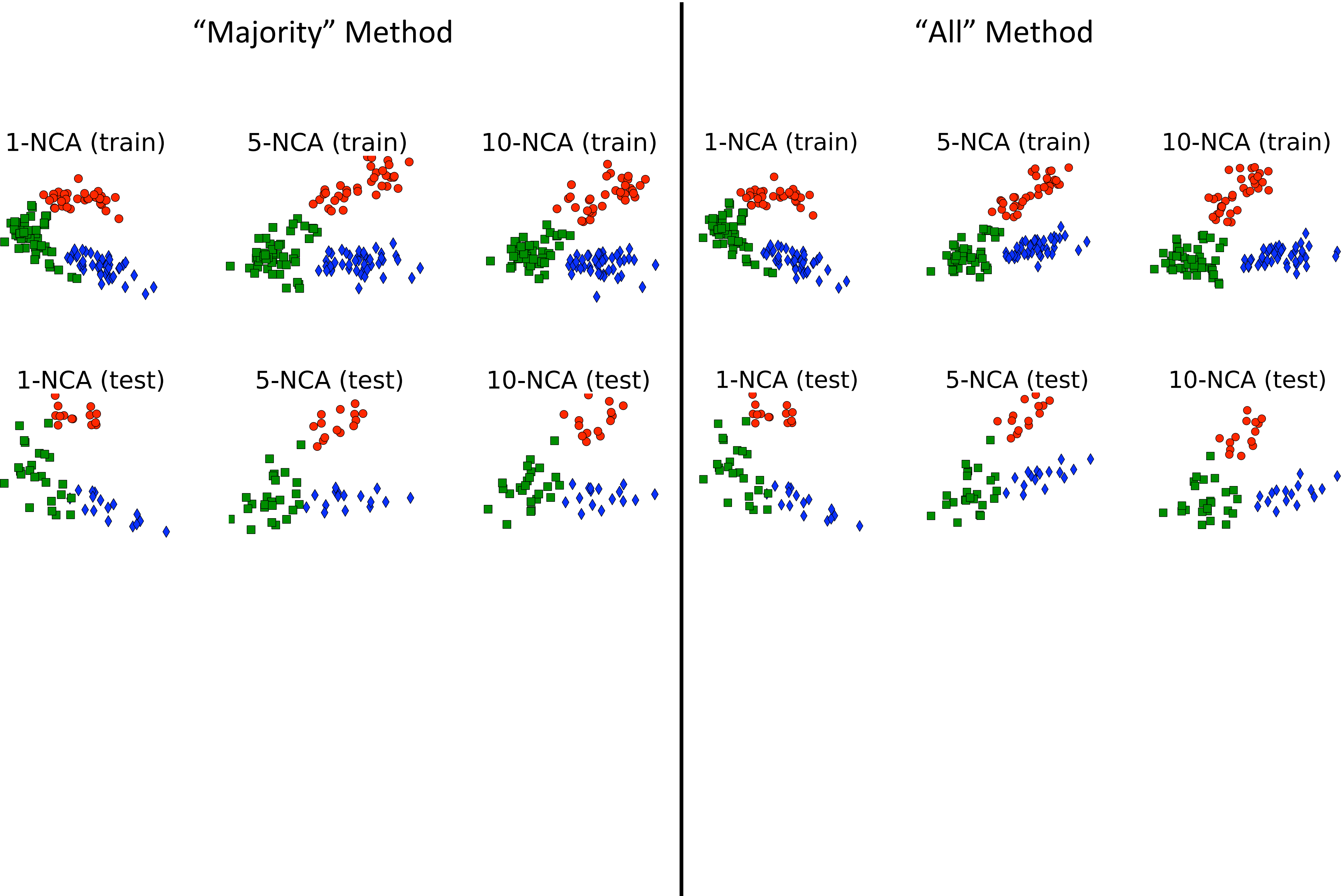
kt-SNE

- kt-SNE can potentially lead to better higher order structure preservation (exponentially many more distance constraints).

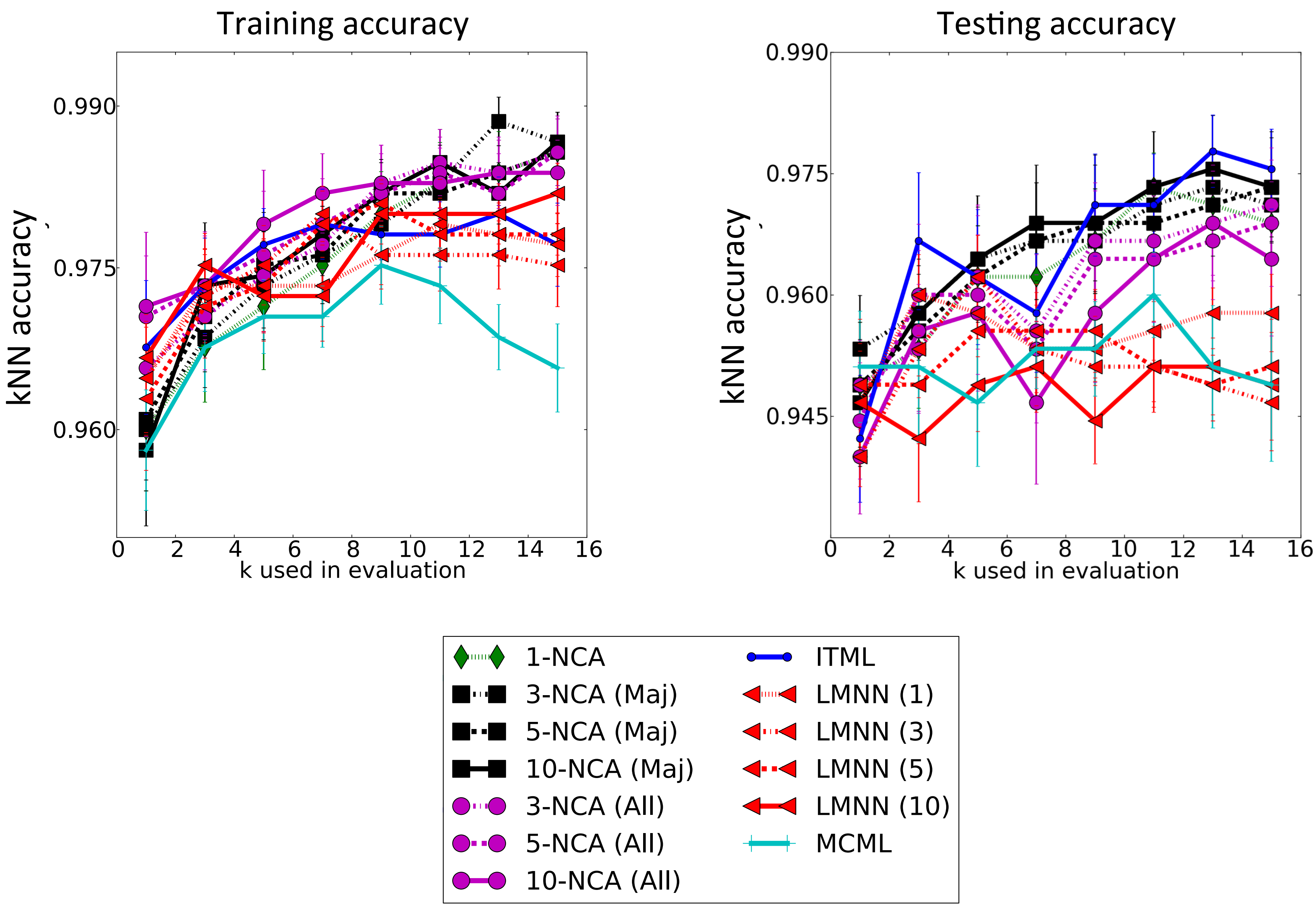
- Gives another “dial to turn” in order to obtain better visualizations.

Experiments

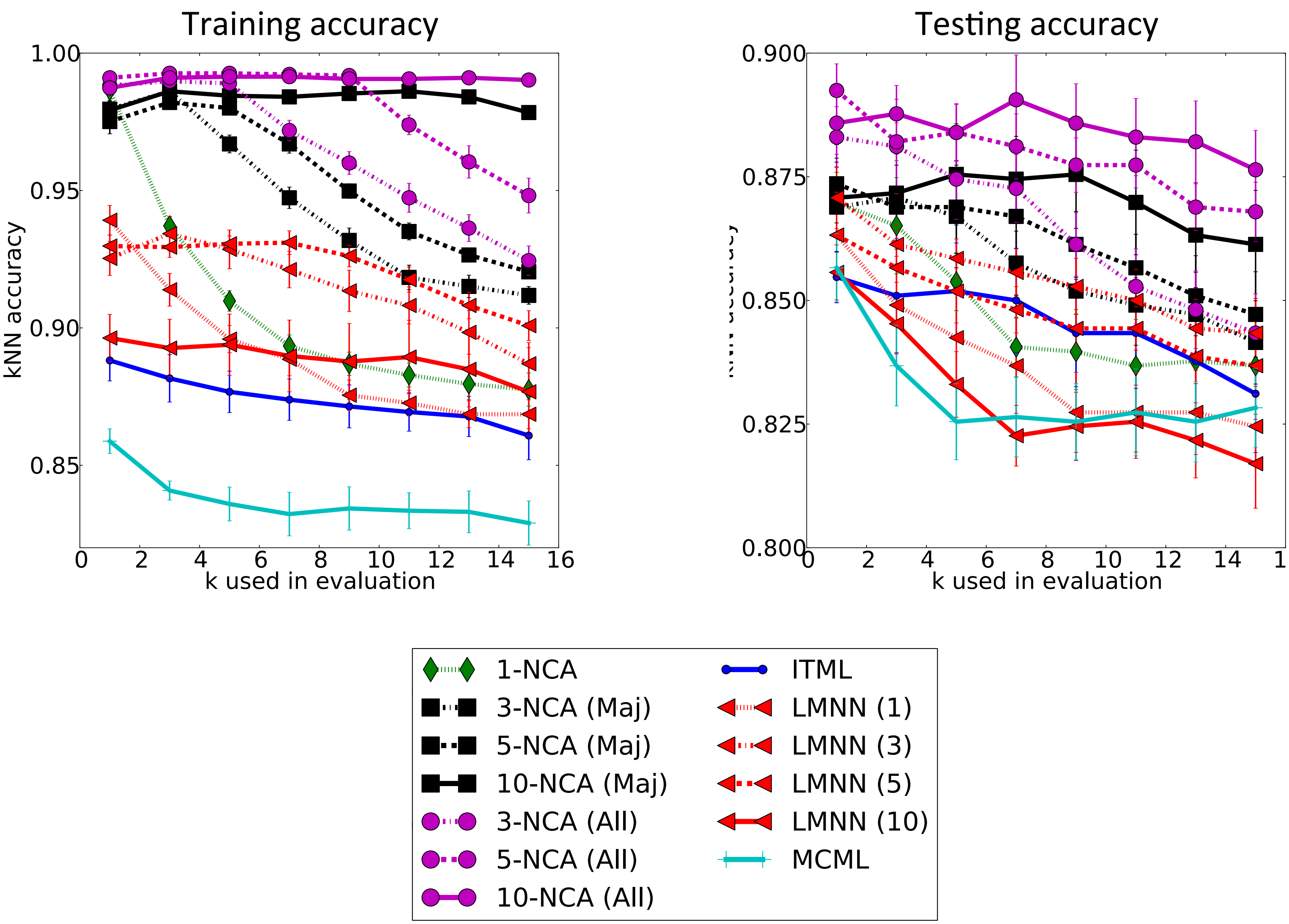
WINE embeddings



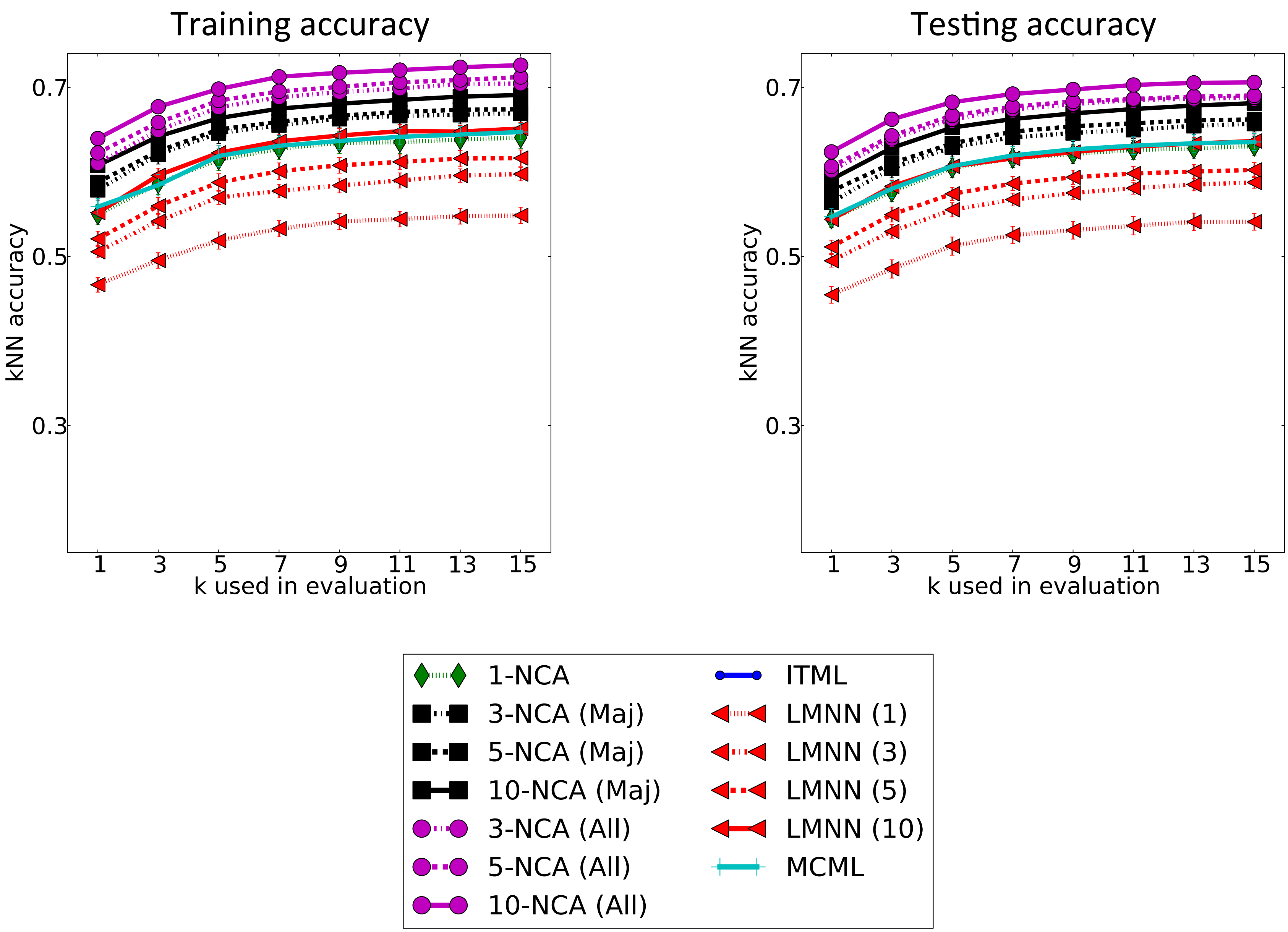
IRIS—worst kNCA relative performance (full D)



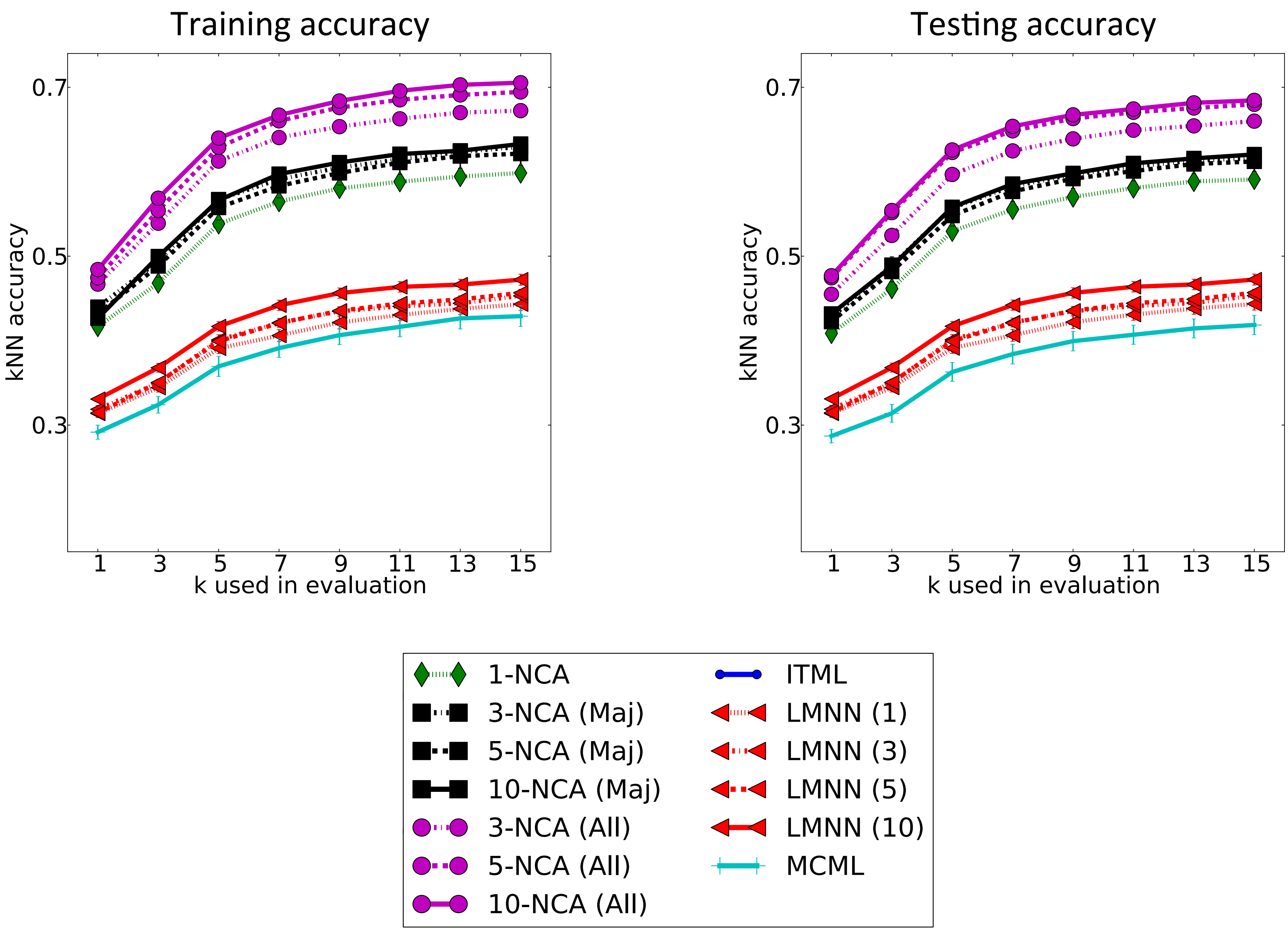
ION—best kNCA relative performance (full D)



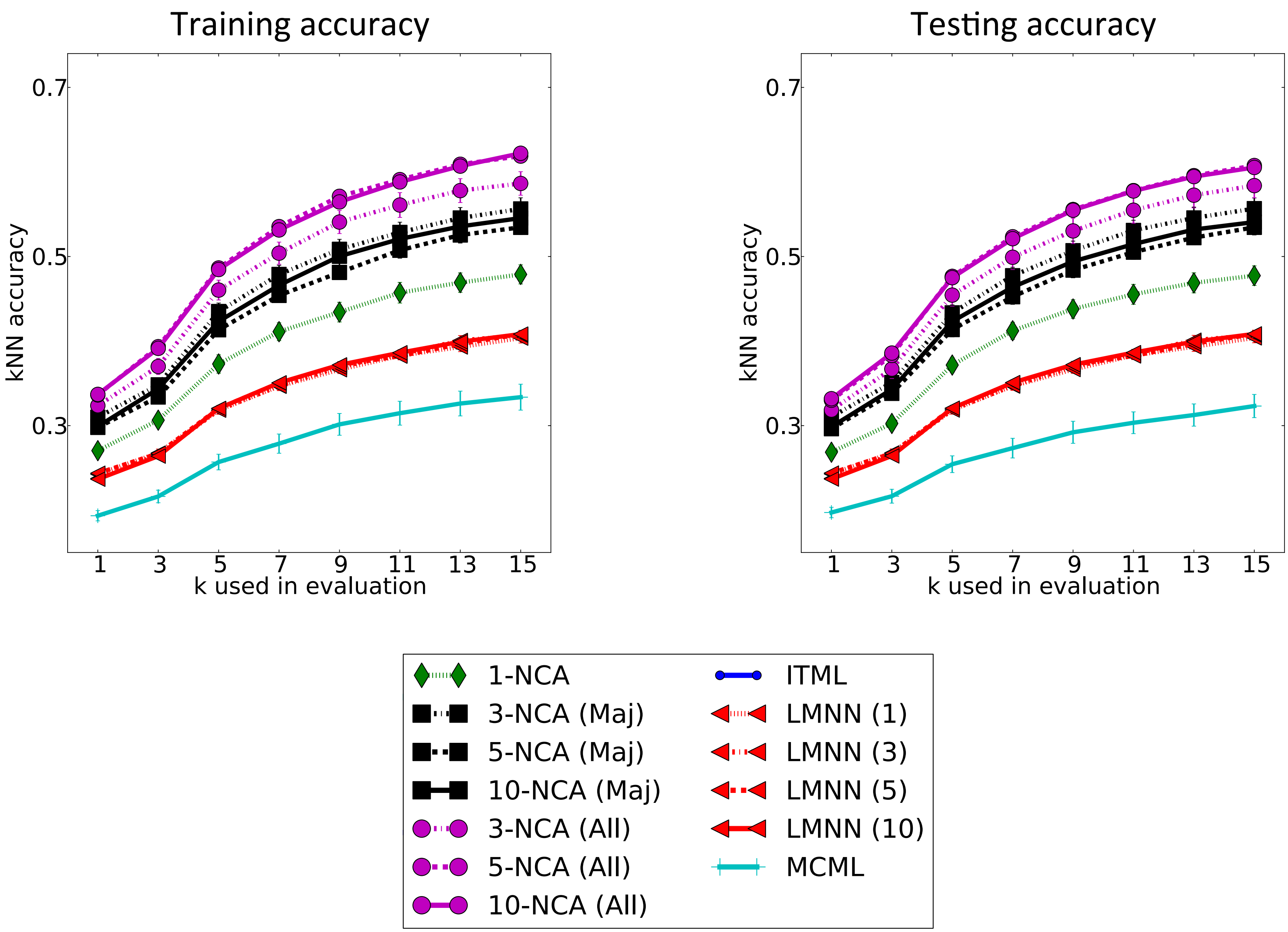
USPS kNN Classification (0% noise, 2D)



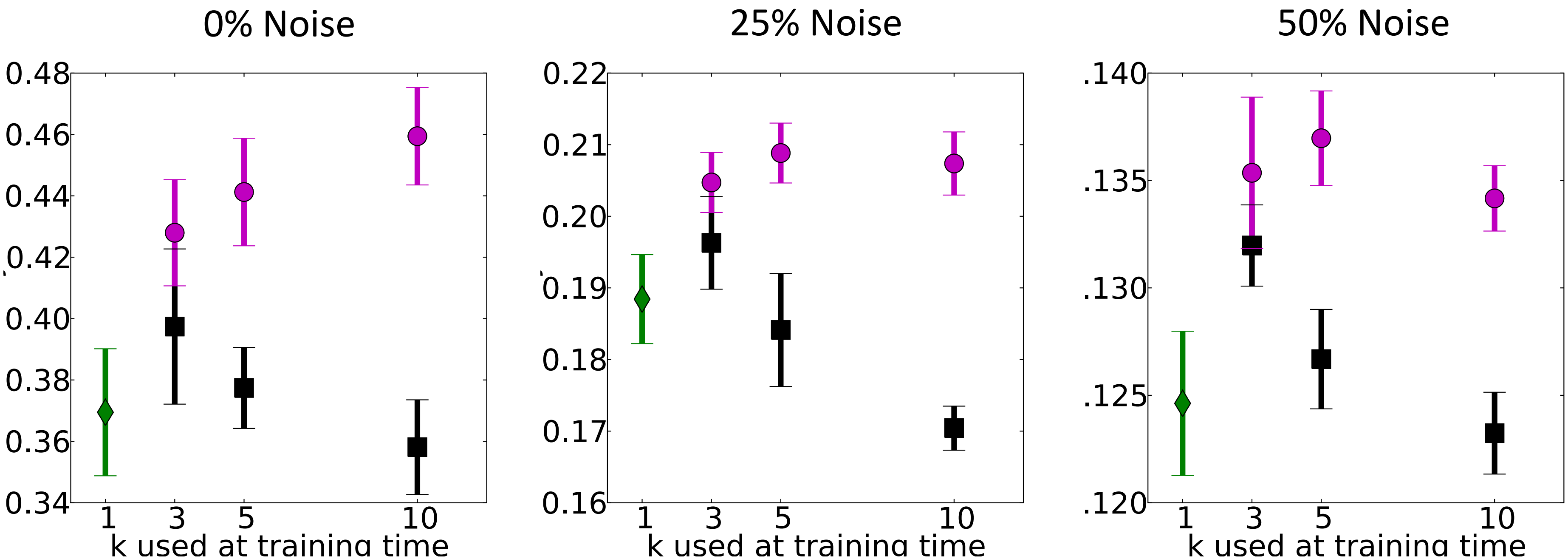
USPS kNN Classification (25% noise, 2D)



USPS kNN Classification (50% noise, 2D)

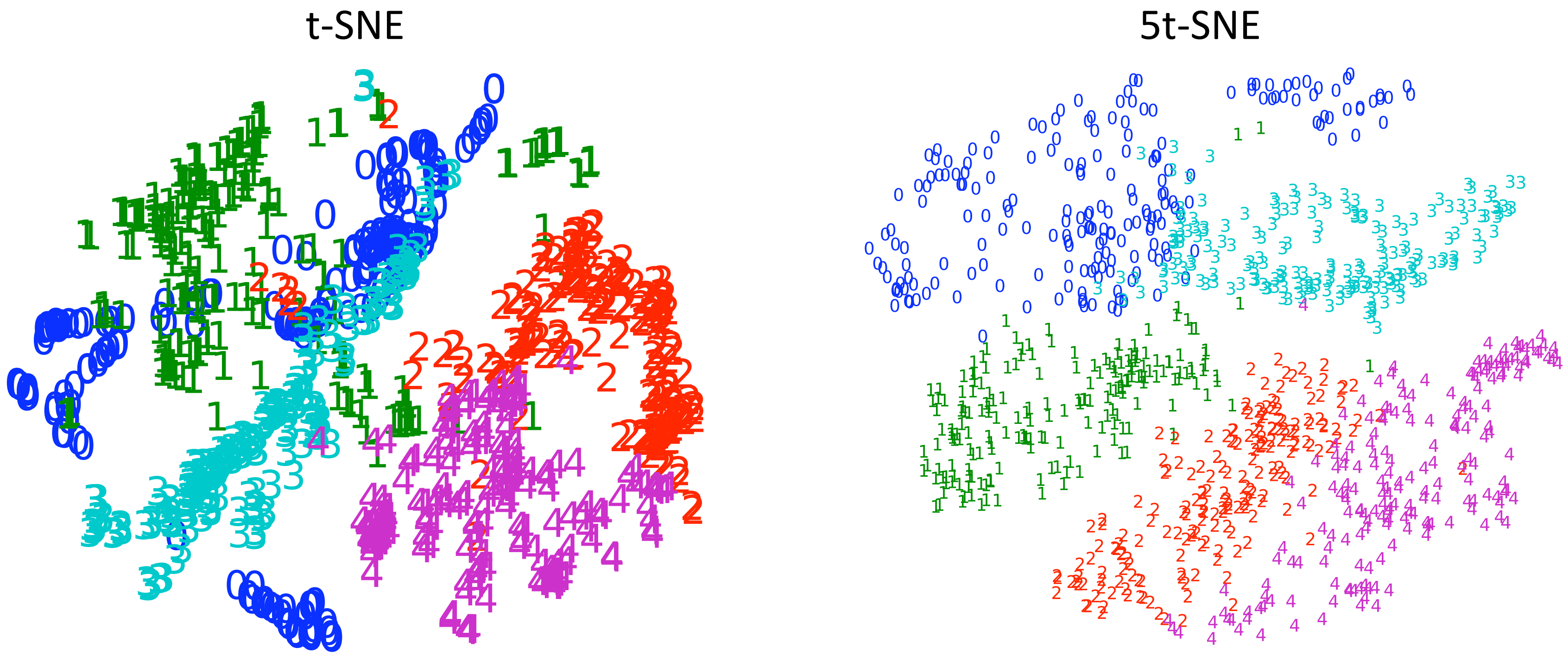


NCA Objective Analysis on Noisy USPS



Y-axis: objective of 1-NCA, evaluated at the parameters learned from k-NCA with varying k and neighbor method

t-SNE vs kt-SNE



kNN Accuracy

	$k=1$	$k=5$	$k=9$	$k=13$
t-SNE	**0.934**	0.925	0.916	0.914
$5t$-SNE	0.928	**0.946**	**0.953**	**0.949**

Discussion

- Local is good, but 1-NCA is too local.
- Not quite expected kNN accuracy, but doesn't seem to change results.
- Expected Majority computation may be useful elsewhere?