A tutorial on Metric Learning with some recent advances

Nakul Verma

HHMI

The Machine Learning Pipeline

Some interesting phenomenon

⇓

Collect various measurements of observations

⇓

Preprocess: data cleaning/curation, and design a more useful representation

⇓

Do machine learning: SVMs, nearest neighbors, perceptrons, decision trees, random forests, etc.

⇓

Try to interpret the results…

The Value of Effective Representation

Discover key factors in data

Encode higher order interactions

Provide simple description of complicated phenomenon

Hand design representations can only get you so far…

perhaps learn a good representation?

This study has created several specialized subfields in machine learning:

Manifold learning, Metric Learning, Deep learning

So What is this Metric Learning?

A type of mechanism to combine features to effectively compare observations

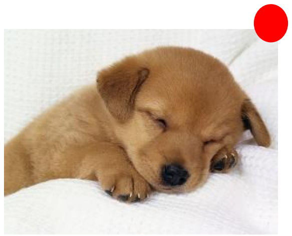
Unsupervised

Principal Component Analysis
Random Projections
IsoMap
Locally Linear Embeddings
Laplacian Eigenmaps
Stochastic Neighbor Embedding

Supervised

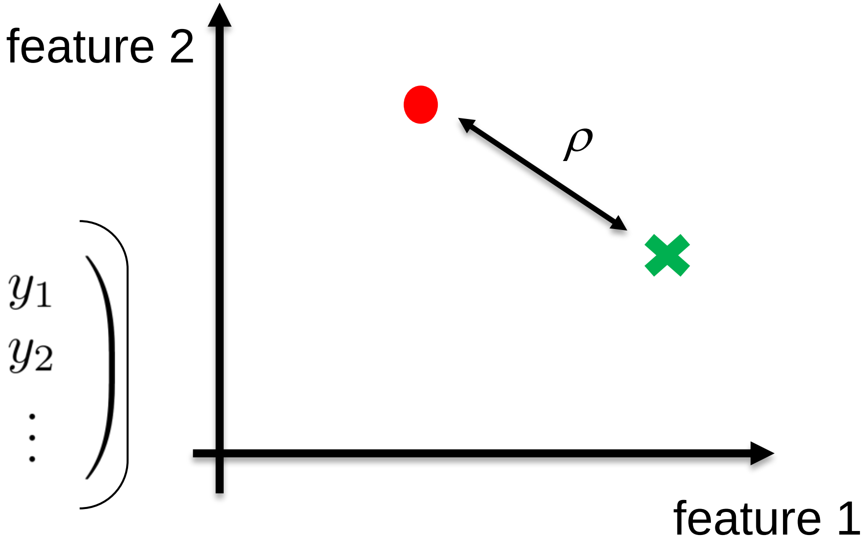
Mahalanobis Metric for Clustering
Large Margin Nearest Neighbor
Neighborhood Component Analysis
Information Theoretic Metric Learning

How to compare observations?



$$\rho(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_d - y_d)^2}$$

$$= \sqrt{\left(\begin{pmatrix} x_1 \\ x_2 \\ \vdots \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \vdots \end{pmatrix} \right) \cdot \left(\begin{pmatrix} x_1 \\ x_2 \\ \vdots \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \vdots \end{pmatrix} \right)}$$



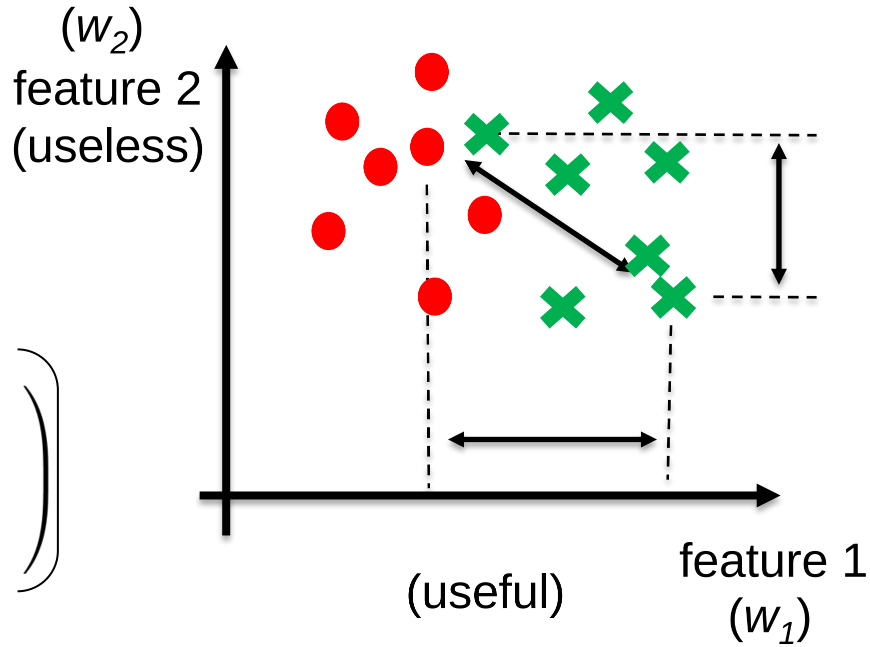
$$= \sqrt{(x - y)^\mathsf{T}(x - y)}$$

But...

Not all features are created equal.

Often some features are noisy or uninformative.

A priori, we don't know which features are relevant for the prediction task at hand.



$$\rho(x, y) = \sqrt{w_1(x_1 - y_1) + w_2(x_2 - y_2)}$$

$$= \sqrt{W\left(\begin{pmatrix} x_1 \\ x_2 \\ \vdots \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \vdots \end{pmatrix}\right) \cdot W\left(\begin{pmatrix} x_1 \\ x_2 \\ \vdots \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \vdots \end{pmatrix}\right)}$$

$$= \sqrt{[W(x-y)]^\top [W(x-y)]} = \sqrt{(x-y)^\top [W^\top W](x-y)} = \sqrt{(x-y)^\top M (x-y)}$$

So what is metric learning?

$$\rho_M(x, y) = \sqrt{(x-y)^\mathsf{T} M (x-y)}$$

Given data of interest, *learn* a *metric* (*M*), which helps in the prediction task.

How?

Want:

Given some annotated data, want to find an M such that examples from the same class get small distance than examples from opposite class.

So:

Create an appropriate optimization problem and optimize for *M*!

The basic optimization

$$\rho_M(x,y) = \sqrt{(x-y)^\mathsf{T} M (x-y)}$$

Attempt: Let's create two sets of pairs: similar set *S*, dissimilar set *D*.

want *M* such that: $\rho_M(x,x')$ large, for $(x,x') \in D$

$\rho_M(x,x')$ small, for $(x,x') \in S$

Create cost/energy function: $\Psi(M)$

$$\Psi(M) = \lambda \sum_{(x,x')\in S} \rho_M^2(x,x') - (1-\lambda) \sum_{(x,x')\in D} \rho_M^2(x,x')$$

Minimize $\Psi(M)$ with respect to *M*!

Detour: How do we minimize?

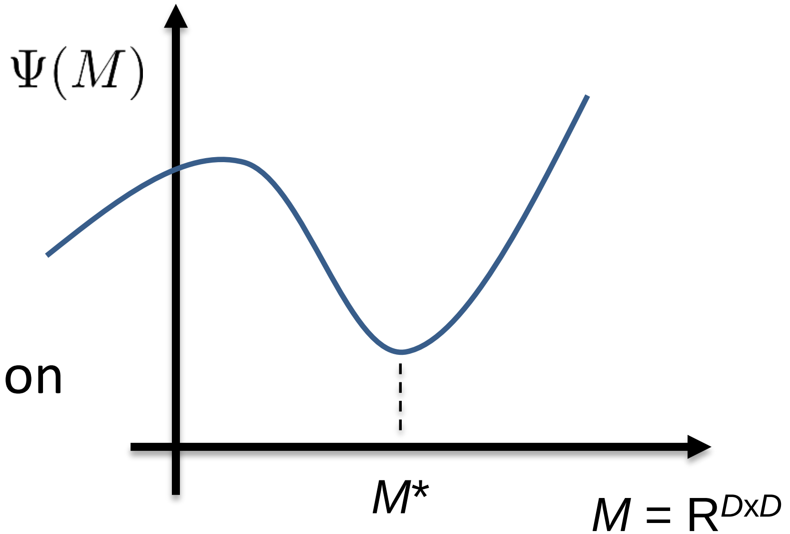
Its an optimization problem!

- Take the gradient
- Find the stationary points

Things to consider:

- There are constraints
- The function is high dimensional

Of course, some certain constraint optimization problems are easier to minimize than others



The basic optimization

Attempt: Let's create two sets of pairs: similar set *S*, dissimilar set *D*.

want M such that: $\rho_M(x, x')$ large, for $(x, x') \in D$

$\rho_M(x, x')$ small, for $(x, x') \in S$

Create cost/energy function: $\Psi(M)$

$$\Psi(M) = \lambda \sum_{(x,x') \in S} \rho_M^2(x, x') \quad - \quad (1 - \lambda) \sum_{(x,x') \in D} \rho_M^2(x, x')$$

Minimize $\Psi(M)$ with respect to *M*!

A Variation… MMC

maximize$_M$ $$\sum_{(x,x')\in D} \rho_M^2(x,x')$$

constraint: $$\sum_{(x,x')\in S} \rho_M^2(x,x') \leq 1$$

$$M \in \text{PSD}$$

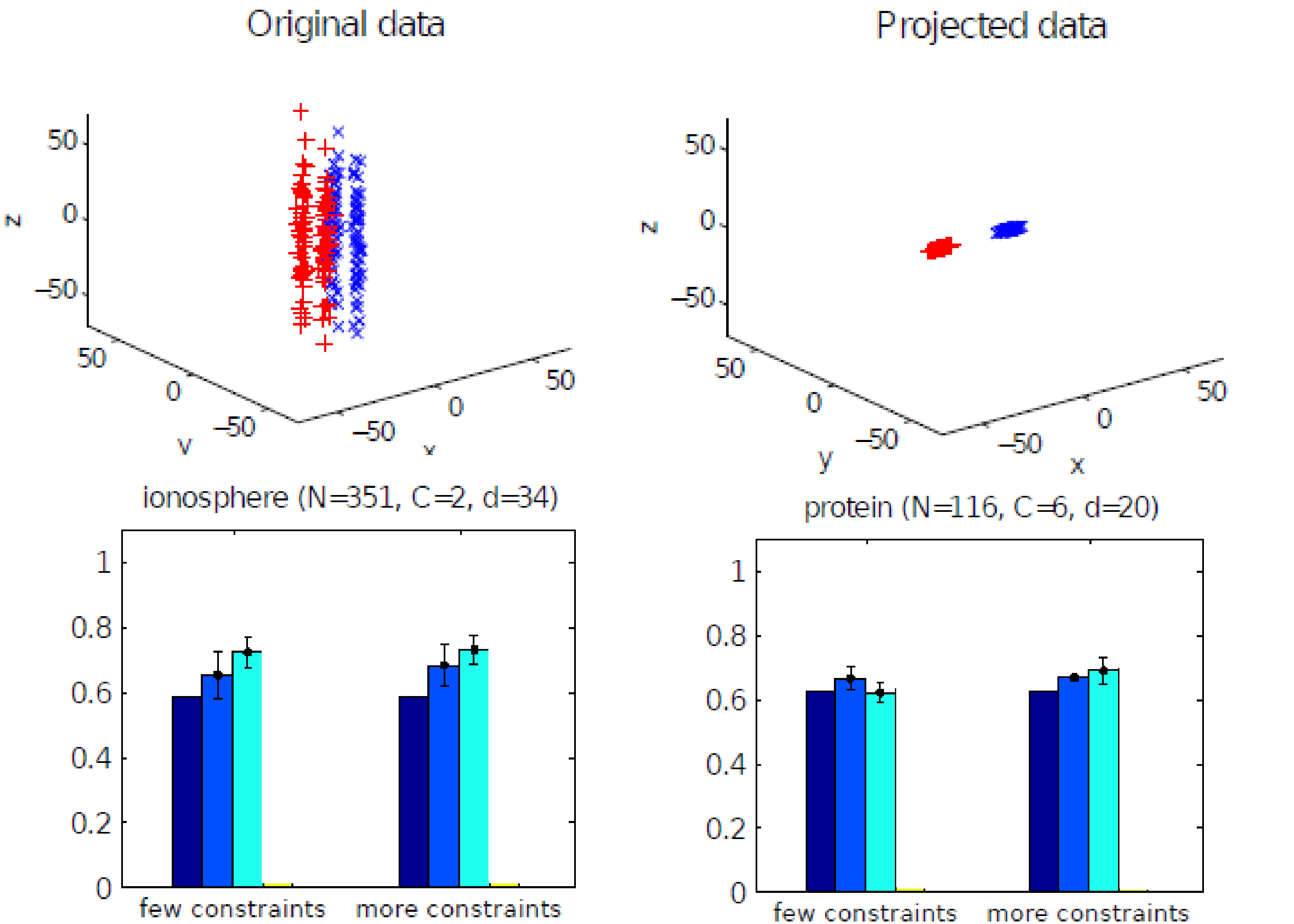
Recall:
$$M = W^\intercal W$$

Advantages:

- Problem formulation is convex, so efficiently solvable!
- Tight convex clusters, can help in clustering!

Xing, Ng, Jordan, Russell, *NIPS 2002.*

Empirical performance



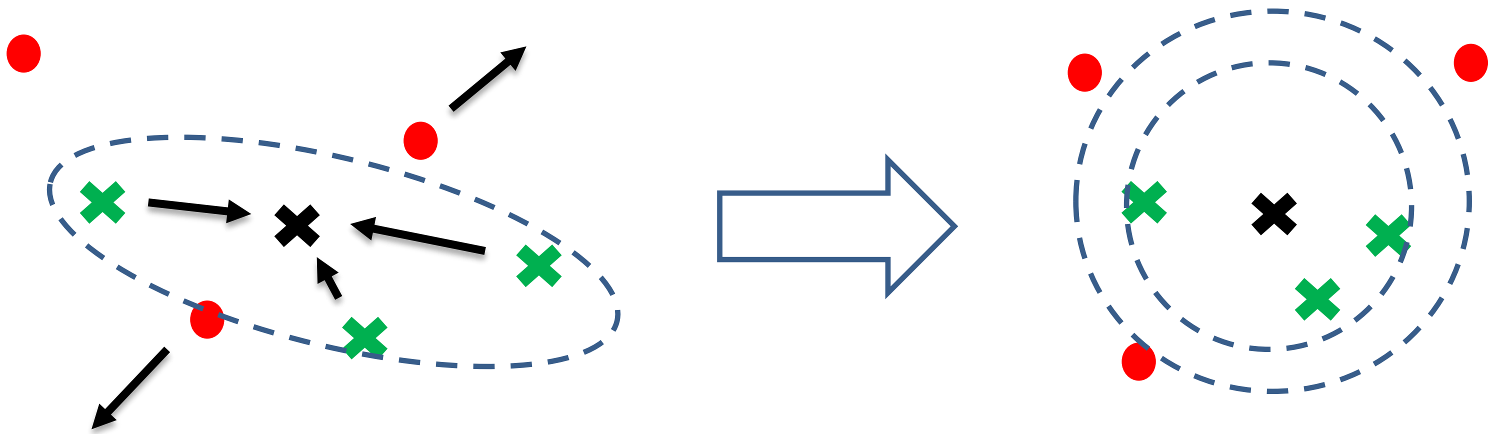
Left to right: *k*-means, *k*-means+diag MMC, *k*-means + full MMC.

Another Interesting Formulation… LMNN

$$\Psi_{\text{pull}}(M) = \sum_{i,j(i)} \rho_M^2(x_i, x_j)$$

$$\Psi_{\text{push}}(M) = \sum_{i,j(i),l(i,j)} 1 + \rho_M^2(x_i, x_j) - \rho_M^2(x_i, x_l)$$

point	i
true neighbor	$j(i)$
imposter	$l(i,j)$



$$\Psi(M) = \lambda \ \Psi_{\text{pull}}(M) \ + (1-\lambda) \ \Psi_{\text{push}}(M)$$

Advantages:

- Local constraints, so directly improves nearest neighbor quality!

Weinberger, Saul, *JMLR 2009.*

Empirical performance

Query

After learning

Original metric

Empirical performance

Dataset	k-NN best	LMNN best	SVM
mnist	2.12	**1.18**	1.20
letters	4.63	**2.67**	3.21
isolet	5.90	**3.40**	**3.40**
yfaces	4.80	**4.05**	15.22
balance	10.82	5.86	**1.92**
wine	2.17	**2.11**	22.24
iris	4.00	3.68	**3.45**

Metric Learning for Multi-class Classification

Observation:

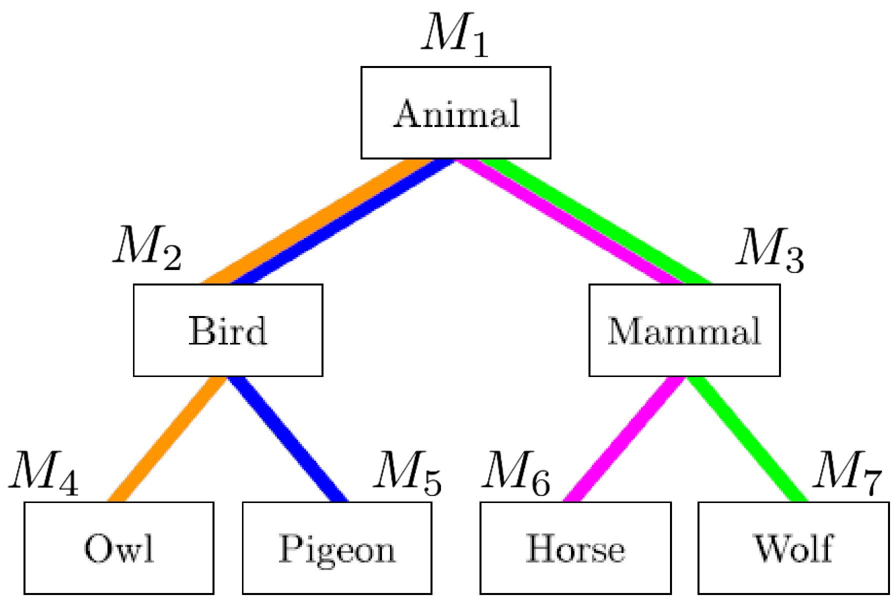
Categories in multiclass data are often part of a underlying ***semantic taxonomy***.

Goal:

To learn ***distance metrics*** that leverage the class taxonomy to yield good classification performance.

$$\mathbf{M}_{\text{owl}} = M_1 + M_2 + M_4$$

$$\mathbf{M}_{\text{horse}} = M_1 + M3 + M_6$$



Metric Learning for Multi-class Classification

Given a query, define its affinity to a class: $f(x_q; y) := \sum_{x \in \mathcal{N}_y(x_q)} \rho(x_q, x; \mathbf{M}_y)$

So, putting it in probabilistic framework:

$$p(y|x, M_1, \ldots, M_T) := \frac{\exp(-f(x; y, \mathbf{M}_y))}{\sum_{\bar{y}} \exp(-f(x; \bar{y}, \mathbf{M}_{\bar{y}}))}$$

Now, given training samples: $(x_1, y_1), \ldots, (x_n, y_n)$
we obtain a good set of metrics $M_1, \ldots, M_T$ by maximizing:

$$\mathcal{L}(M_1, \ldots, M_T) := \frac{1}{n} \sum_{i=1}^{n} \log p(y_i | x_i; M_1, \ldots, M_T) - \frac{\lambda}{2} \sum_t \operatorname{trace}(M_t^{\mathsf{T}} M_t)$$

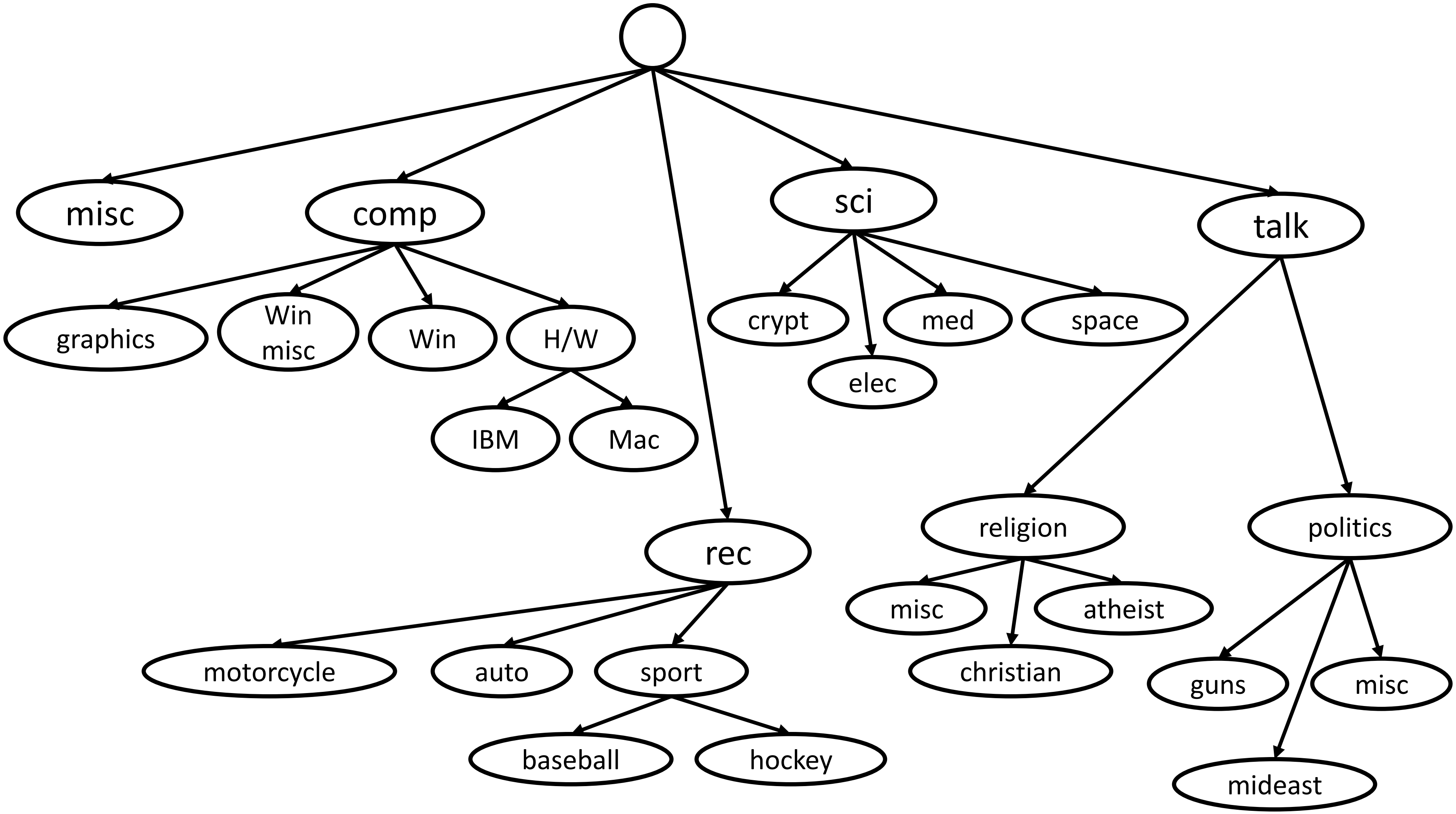
Observations:

- Optimization is jointly **convex**.
- Geometrically, the likelihood is maximized by: ***pulling together*** the neighbors belonging to the same class, while ***pushing away*** the neighbors from different class.

V., Mahajan, Sellamanikam, Nair, *CVPR 2012.*

Empirical performance

20 Newsgroup dataset – 20 classes, with 20k articles.



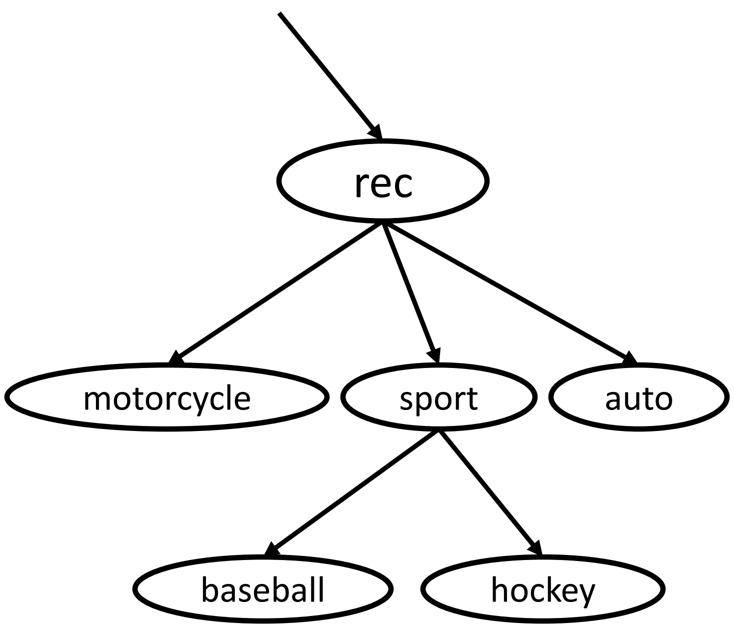
Empirical performance

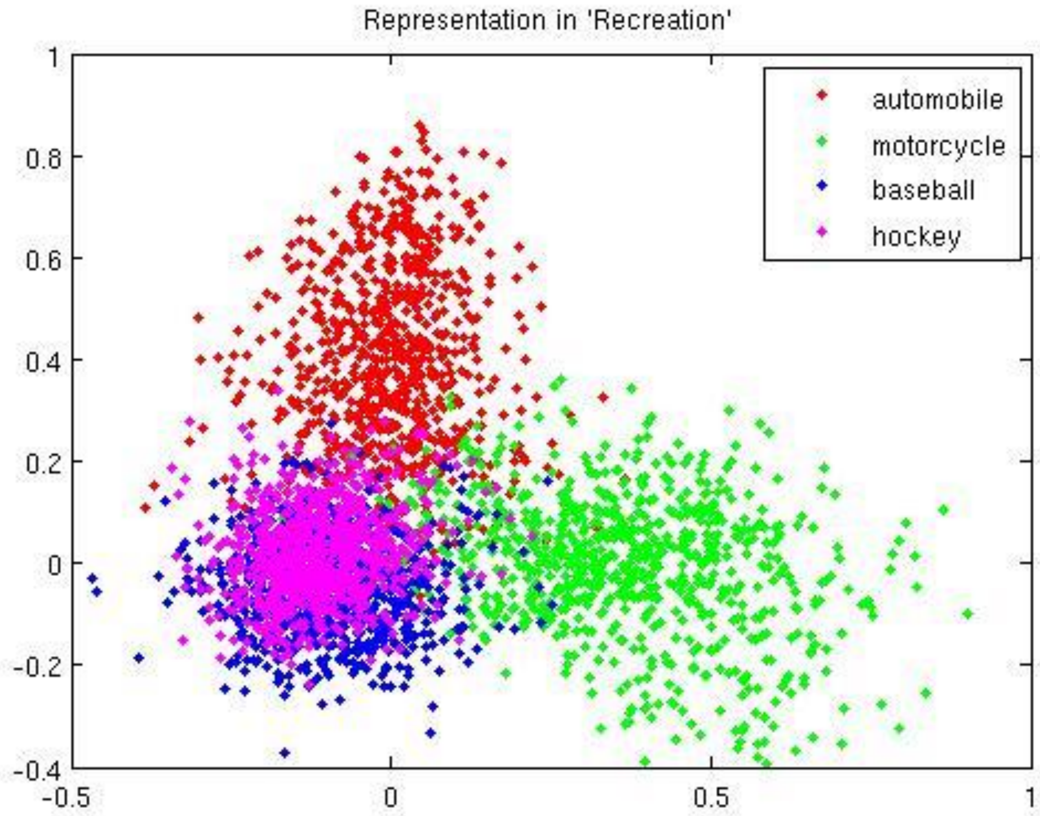
Performance (accuracy)

20 Newsgroups	SVM	Euclid	Flat	Hierarchy
Train Data (16k)	0.86	0.80	0.87	**0.94**
Test Data (4k)	0.80	0.79	0.80	**0.85**

Empirical performance

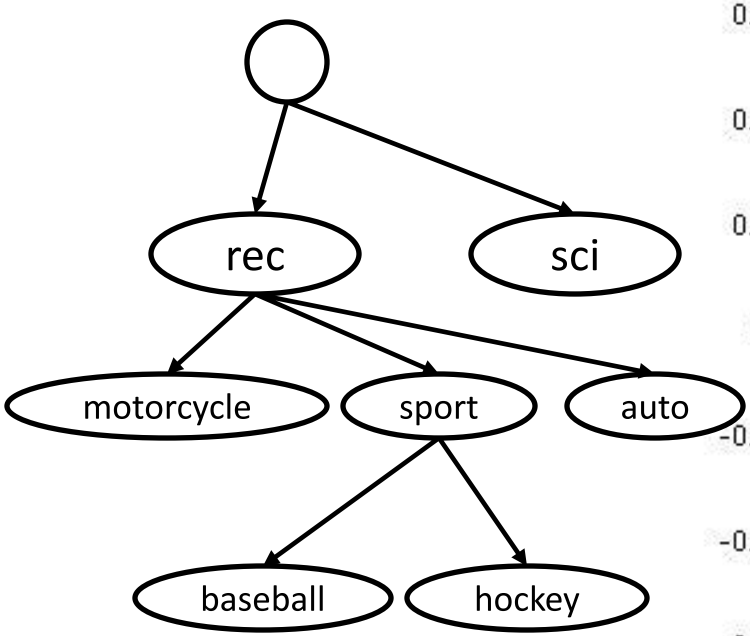
Visualizations:

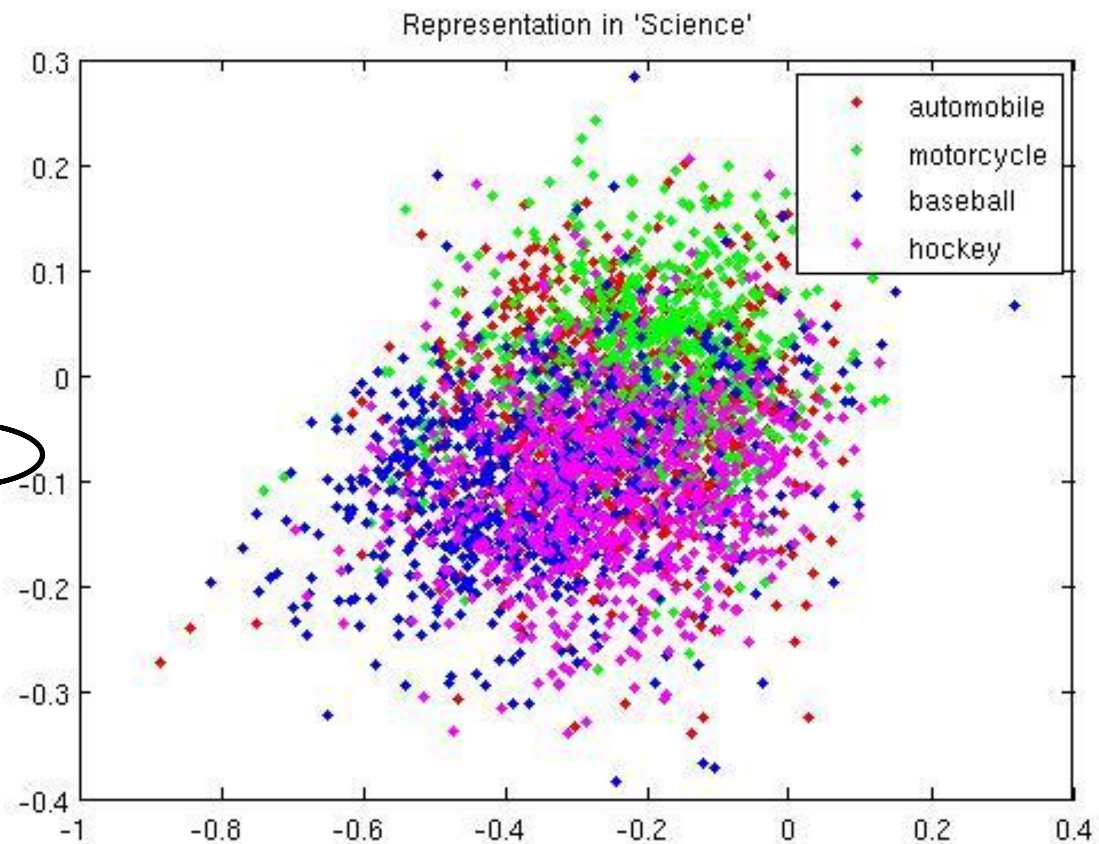




Empirical performance

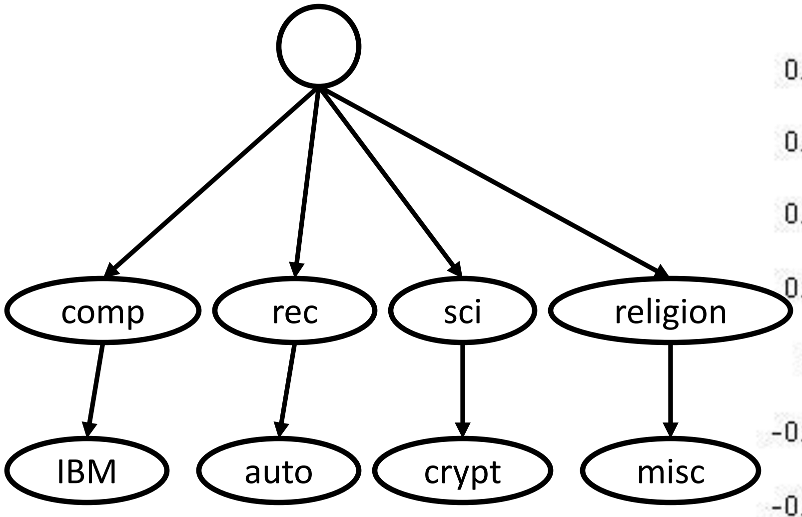
Visualizations:

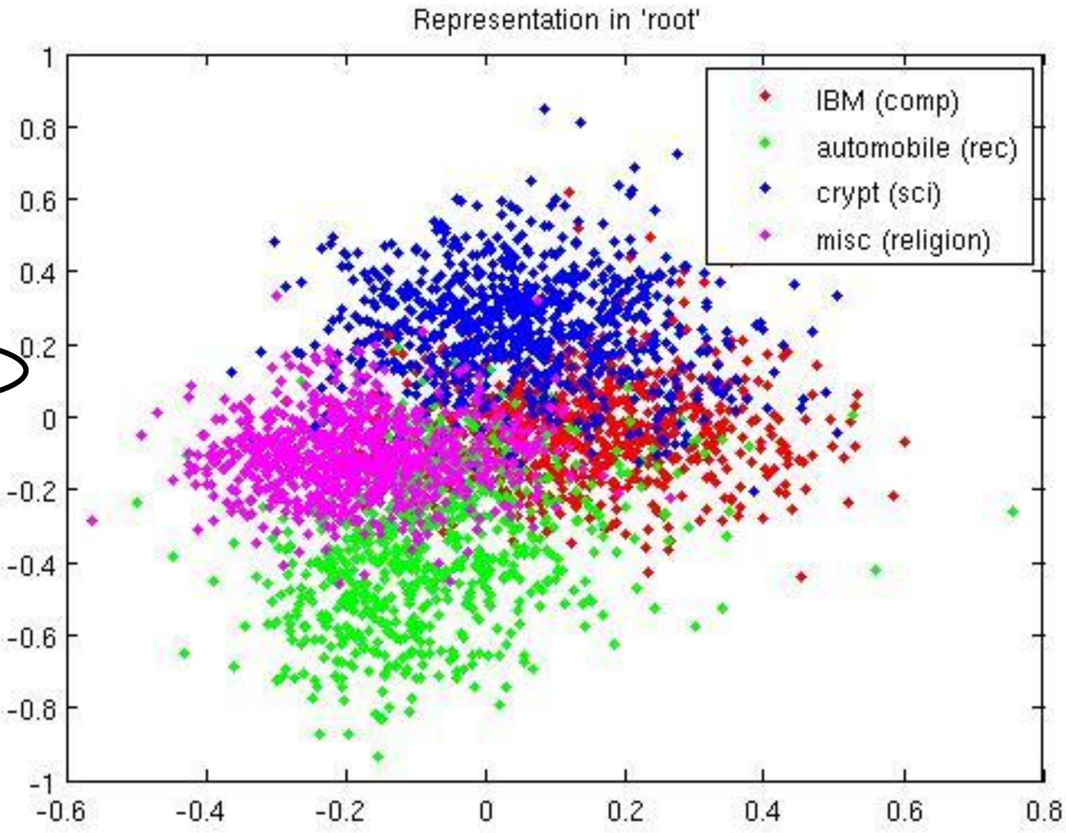




Empirical performance

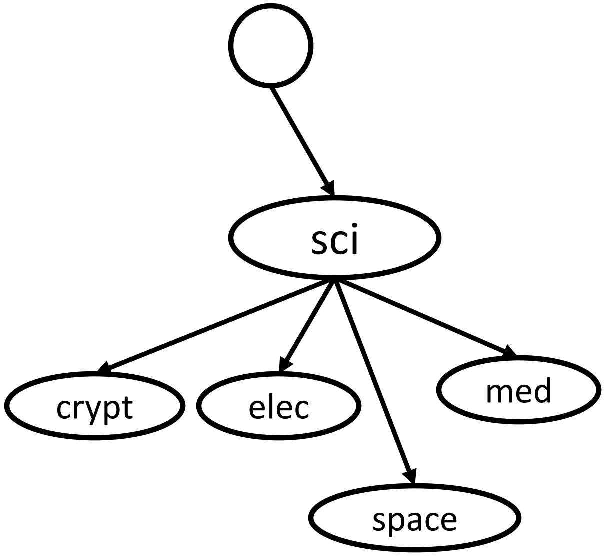
Visualizations:

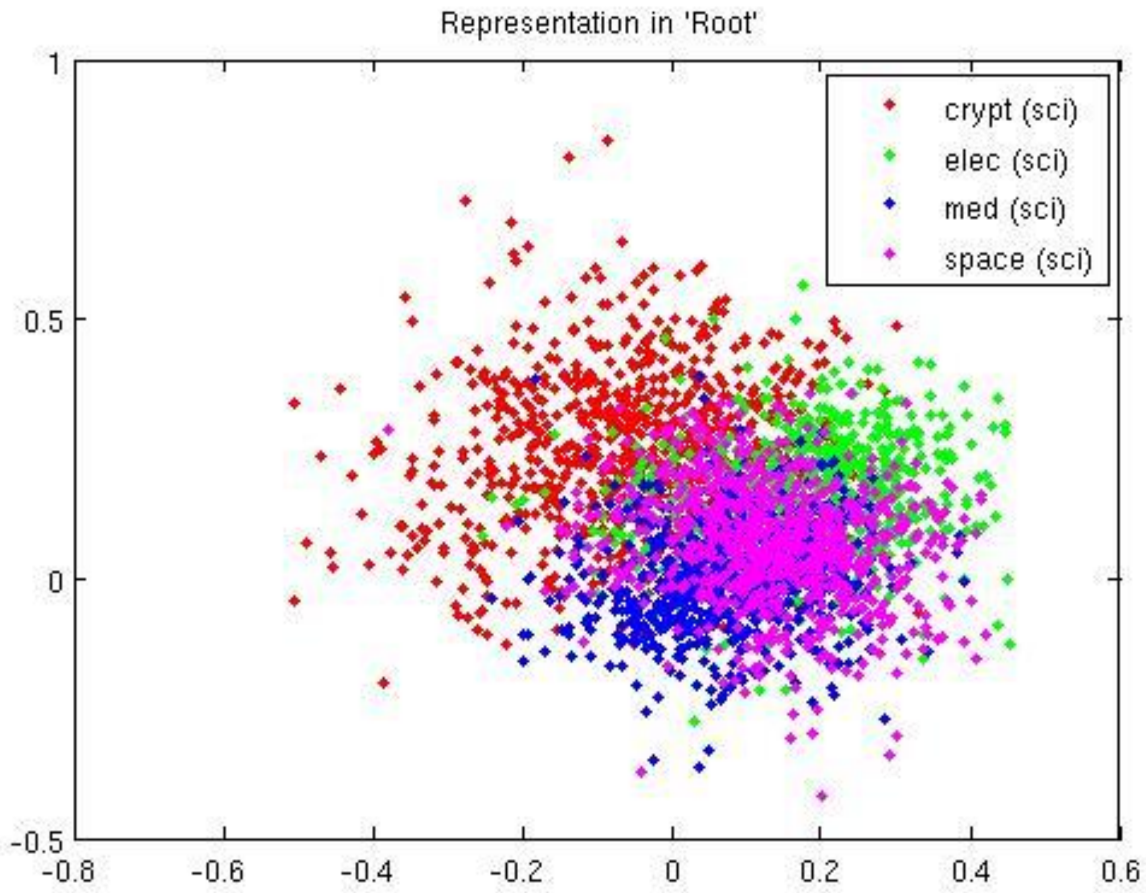




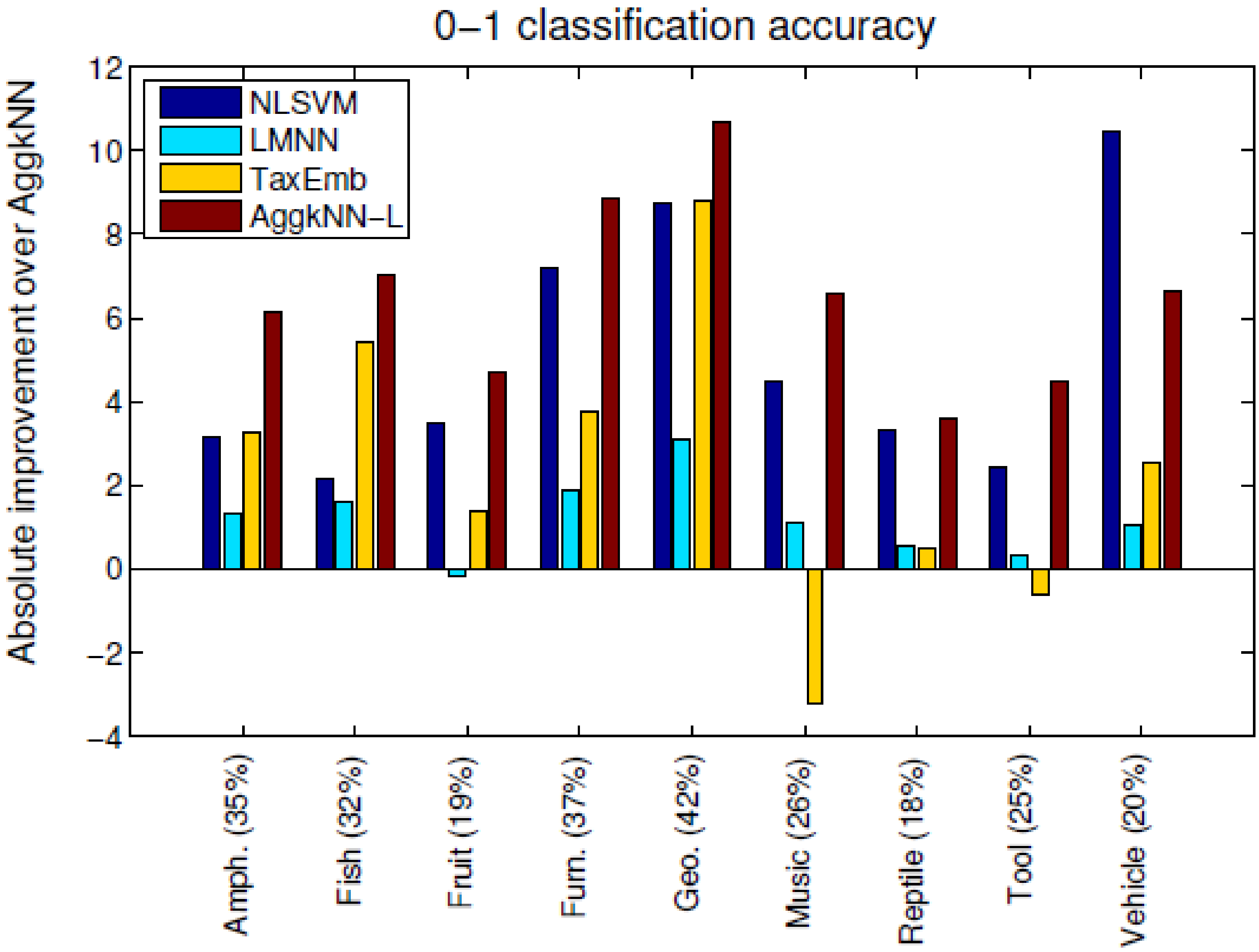
Empirical performance

Visualizations:





Empirical performance



Metric Learning for Information Retrieval

Problem:

Information retrieval: find most relevant examples for a given query.

Goal:

Learn ***distance metric*** that can rank the examples in a database effectively.

Observations:

Output has a structure (ranking) associated with it.

$q \in \mathcal{X}$	$y \in \mathcal{Y}$	$y_q^* \in \mathcal{Y}$	$\psi(q, y)$
(input space)	(output space)	(optimal output)	(joint feature space)

$$\text{minimize}_w \quad \sum_{q \in \mathcal{X}} \xi_q \quad + \quad \lambda \;\; \mathrm{reg}(w)$$

$$\underbrace{\langle w, \psi(q, y_q^*) \rangle}_{\text{score(good ranking)}} \quad \geq \quad \underbrace{\langle w, \psi(q, y) \rangle}_{\text{score(bad ranking)}} \quad + \quad \underbrace{\Delta(y_q^*, y)}_{\text{loss(bad ranking)}} \quad - \quad \xi_q$$

Metric Learning for Information Retrieval

Feature representation for rankings

$$\psi(q, y) = \sum_{i \in \mathcal{X}_q^+} \sum_{j \in \mathcal{X}_q^-} a_{i,j} \frac{\phi(q,i) - \phi(q,j)}{|\mathcal{X}_q^+||\mathcal{X}_q^-|} \qquad a_{i,j} = \begin{cases} +1 & \text{if } i \text{ before } j \\ -1 & \text{if } i \text{ after } j \end{cases}$$

$$\phi(i, j) = (q - i)(q - i)^\mathsf{T}$$

why does this work? $\rho_M^2(x, y) = (x - y)^\mathsf{T} M (x - y) = \langle M, (x - y)(x - y)^\mathsf{T} \rangle_F$

$$\text{minimize}_M \quad \sum\nolimits_{q \in \mathcal{X}} \xi_q + \lambda \ \mathrm{trace}(M)$$

$$\langle M, \psi(q, y_q^*) \rangle_F \geq \langle M, \psi(q, y) \rangle_F + \Delta(y_q^*, y) - \xi_q$$

score(good ranking) score(bad ranking) loss(bad ranking)

McFee, Lanckriet, *ICML 2010.*

Empirical performance

eHarmony dataset: ~250k users, ~450k matchings, feature representation in R^{56}

eHarmony	SVM-MAP	Euclidean	MLR-MAP
AUC	0.614	0.522	**0.624**
MAP	0.447	0.394	**0.453**
MRR	0.467	0.414	**0.474**

The "Theory" of Metric Learning

How hard/easy is it to learn *M* as a function of key properties of data?

- Presence of uninformative features or noisy features?
- How does dimension plays into the sample complexity of learning?

Key result (work in progress)

Theorem: For all data distributions in R^D, given *m* random samples from it:

$$err(M^*) \quad \leq \quad err(\widehat{M}_m) \;+\quad O(D/\sqrt{m})$$

Theorem: for all data distributions in R^D with d relevant features, given *m* random samples from it:

$$err(M^*) \quad \leq \quad err(\widehat{M}_m) \;+\quad O(d \log D/\sqrt{m})$$

V., *In preparation 2014.*

Summary

- Metric Learning:

  A powerful technique to combine features for effective comparison between observations.

- The basic technique has been extended for multiple learning problems

  Large multi-class classification, information retrieval, multi-task learning, domain adaptation, semi-supervised learning

- Interesting questions:

  Adapting to changing data, account for multiple ways to compare data, incorporating more sophisticated structure/geometry into account.

Questions / Discussion

Thank You!

SFML-MG (Tony and David)

Flurry: for hosting the event

The Audience: for patiently listening!