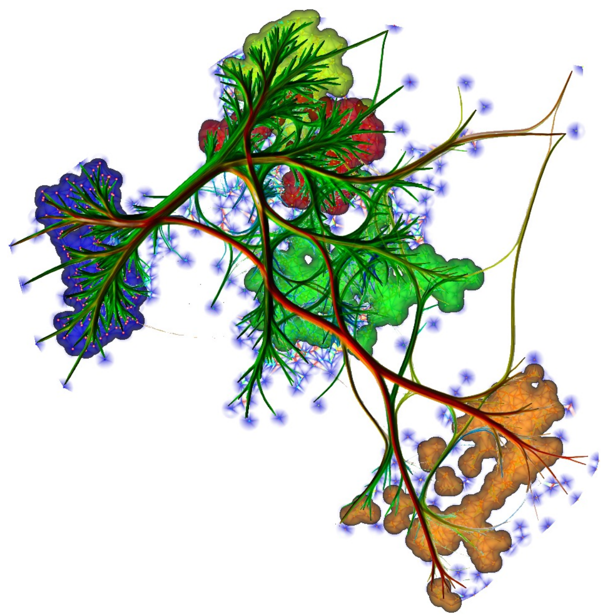
Beyond the Third Dimension

How Multidimensional Projections and Machine Learning Can Help Each Other



prof. dr. Alexandru (Alex) Telea

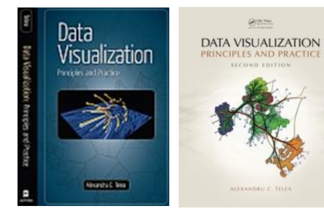
Department of Information and Computing Science
Utrecht University, the Netherlands

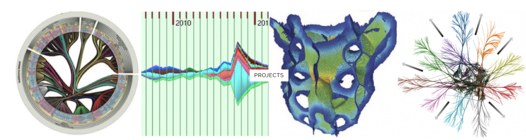
Utrecht University

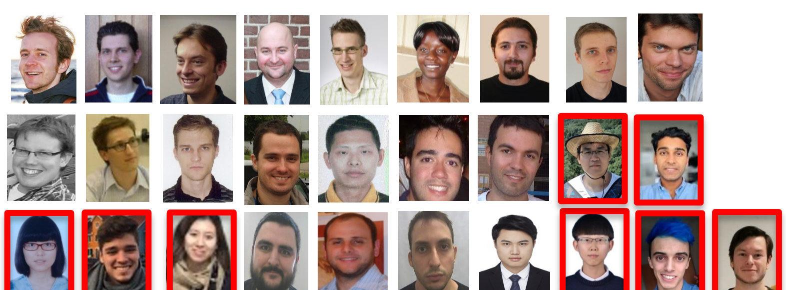
Alexandru Telea

Professor Visual Data Analytics, Utrecht University, Netherlands

- full professor (UU, since 2019; RUG 2007-2019)
- 30 PhD students
- group leader Visualization and Graphics







webspace.science.uu.nl/~telea001

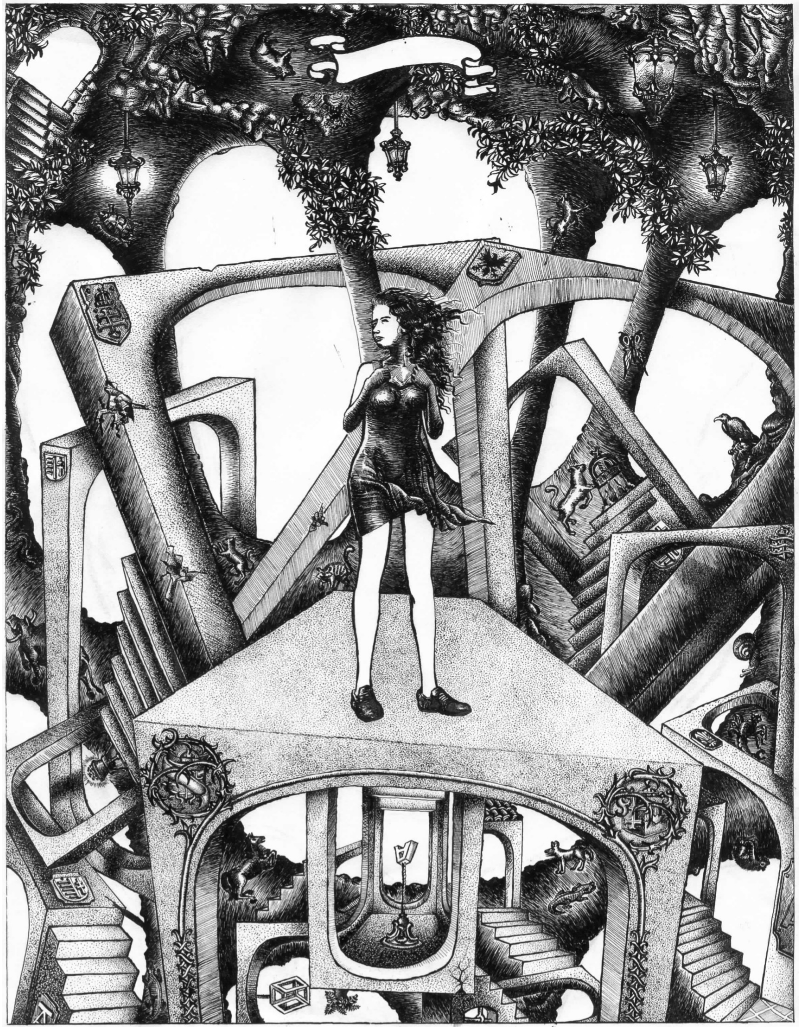
a.c.telea@uu.nl

vig.science.uu.nl

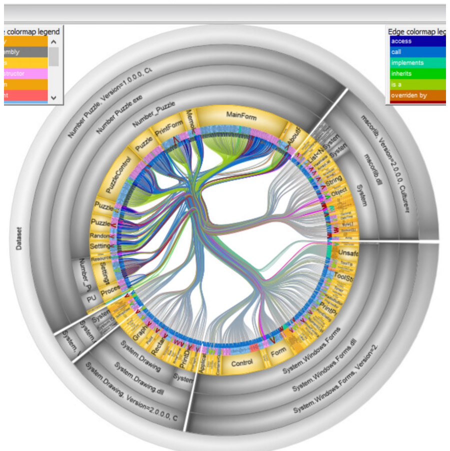
A Bit of (Personal) History

before 1980

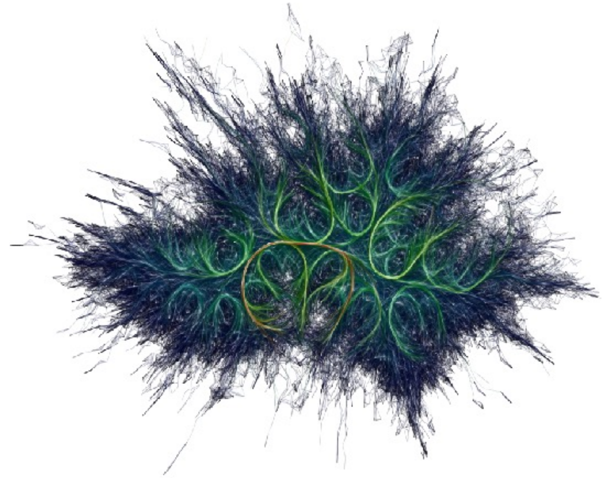
around 2000



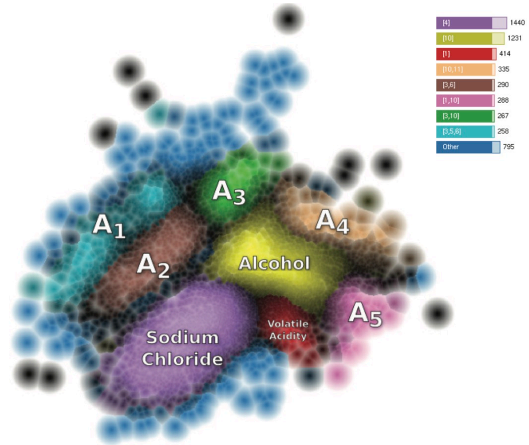
Current Research



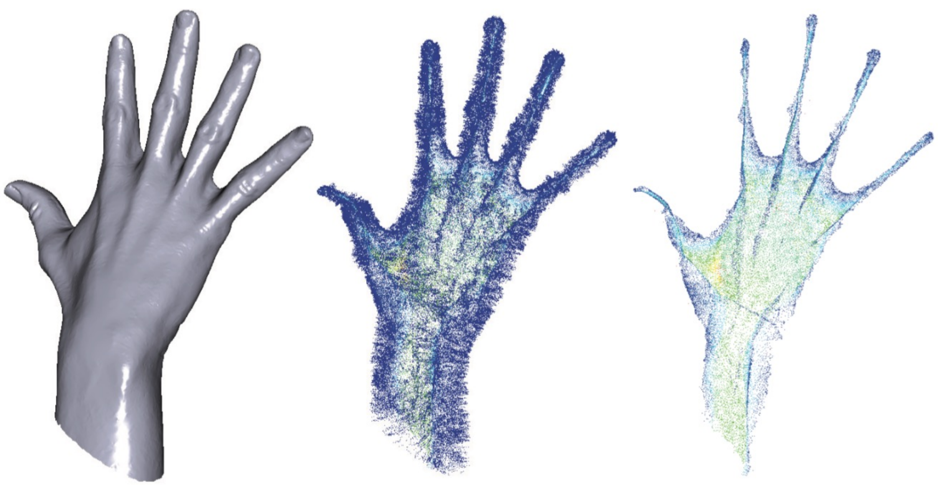
Software visualization



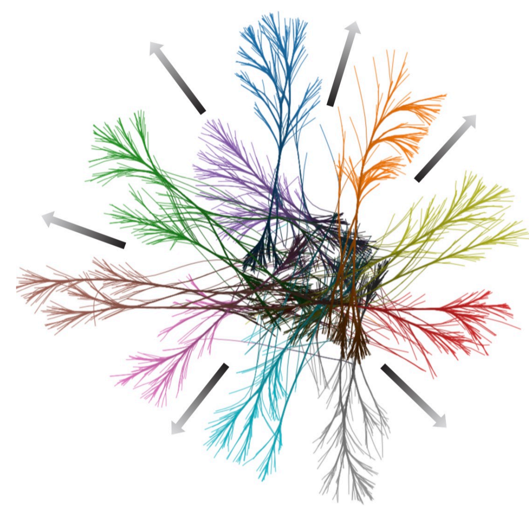
Simplified network visualization



High-dimensional data



Multiscale shape processing

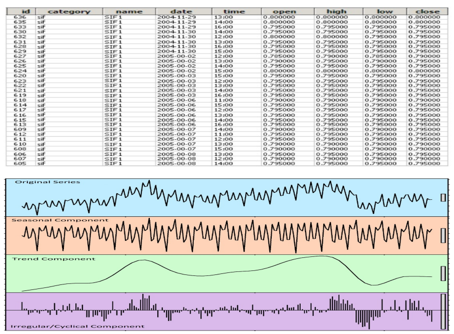


Visual analytics for machine learning

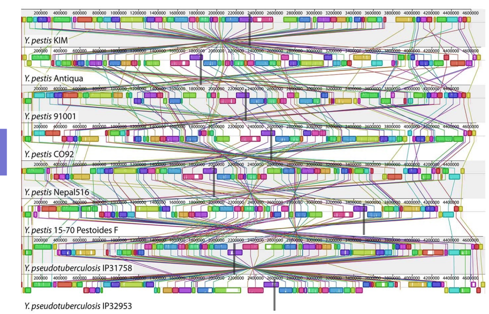
What is *really high-dimensional* data?

Hundreds of dimensions with often no clear meaning

High-dimensional data

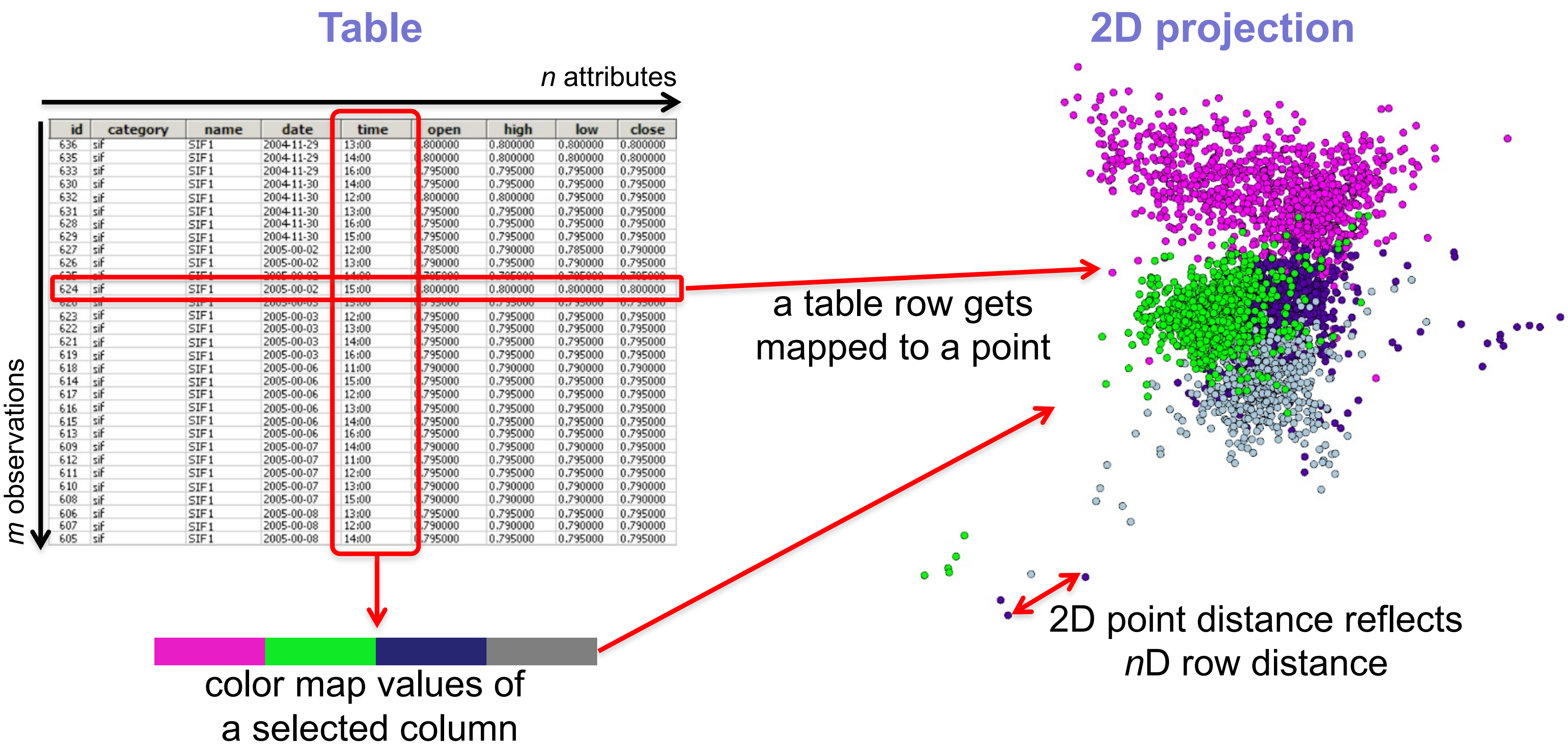


Machine Learning



Solution: Multidimensional projections

Projections

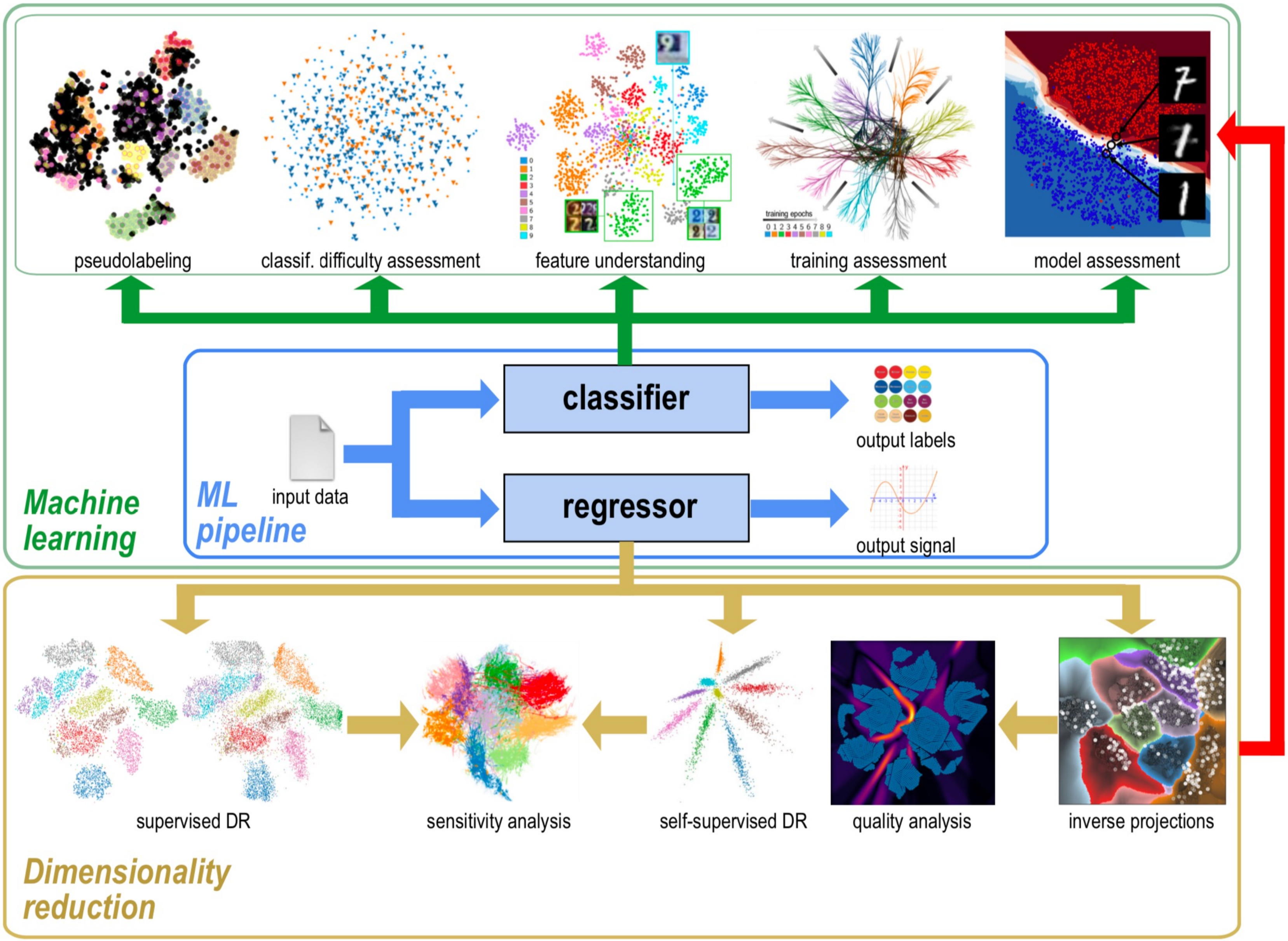


id	category	name	date	time	open	high	low	close
636	sif	SIF1	2004-11-29	13:00	.800000	0.800000	0.800000	0.800000
635	sif	SIF1	2004-11-29	14:00	.800000	0.800000	0.800000	0.800000
633	sif	SIF1	2004-11-29	16:00	.795000	0.795000	0.795000	0.795000
630	sif	SIF1	2004-11-30	14:00	.795000	0.795000	0.795000	0.795000
632	sif	SIF1	2004-11-30	12:00	.800000	0.800000	0.795000	0.795000
631	sif	SIF1	2004-11-30	13:00	.795000	0.795000	0.795000	0.795000
628	sif	SIF1	2004-11-30	16:00	.795000	0.795000	0.795000	0.795000
629	sif	SIF1	2004-11-30	15:00	.795000	0.795000	0.795000	0.795000
627	sif	SIF1	2005-00-02	12:00	.785000	0.790000	0.785000	0.790000
626	sif	SIF1	2005-00-02	13:00	.790000	0.795000	0.790000	0.795000
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
624	sif	SIF1	2005-00-02	15:00	.800000	0.800000	0.800000	0.800000
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
623	sif	SIF1	2005-00-03	12:00	.795000	0.795000	0.795000	0.795000
622	sif	SIF1	2005-00-03	13:00	.795000	0.795000	0.795000	0.795000
621	sif	SIF1	2005-00-03	14:00	.795000	0.795000	0.795000	0.795000
619	sif	SIF1	2005-00-03	16:00	.795000	0.795000	0.795000	0.795000
618	sif	SIF1	2005-00-06	11:00	.790000	0.790000	0.790000	0.790000
614	sif	SIF1	2005-00-06	15:00	.795000	0.795000	0.795000	0.795000
617	sif	SIF1	2005-00-06	12:00	.795000	0.795000	0.795000	0.795000
616	sif	SIF1	2005-00-06	13:00	.795000	0.795000	0.795000	0.795000
615	sif	SIF1	2005-00-06	14:00	.795000	0.795000	0.795000	0.795000
613	sif	SIF1	2005-00-06	16:00	.795000	0.795000	0.795000	0.795000
609	sif	SIF1	2005-00-07	14:00	.790000	0.795000	0.790000	0.795000
612	sif	SIF1	2005-00-07	11:00	.795000	0.795000	0.795000	0.795000
611	sif	SIF1	2005-00-07	12:00	.795000	0.795000	0.795000	0.795000
610	sif	SIF1	2005-00-07	13:00	.790000	0.790000	0.790000	0.790000
608	sif	SIF1	2005-00-07	15:00	.790000	0.790000	0.790000	0.790000
606	sif	SIF1	2005-00-08	13:00	.795000	0.795000	0.795000	0.795000
607	sif	SIF1	2005-00-08	12:00	.790000	0.790000	0.790000	0.790000
605	sif	SIF1	2005-00-08	14:00	.795000	0.795000	0.795000	0.795000

Why is this useful?

- no matter how large *n* is, we obtain a 2D scatterplot-like image (so it's visually scalable)
- point-to-point distance (in 2D) shows similarity of observations (in *n*D)
- coloring points by one attribute can show additional information on the observations

Machine Learning and Dimensionality Reduction Commonalities

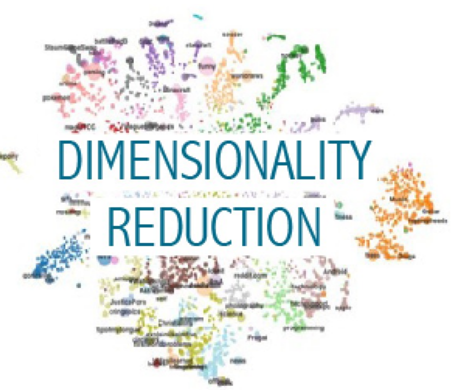


Machine Learning and Dimensionality Reduction Commonalities

Availability

Genericity

Out of Sample



Accuracy

Scalability

Stability

Machine Learning and Dimensionality Reduction Commonalities

Availability

Genericity

Approach ML and DR jointly

Out of Sample

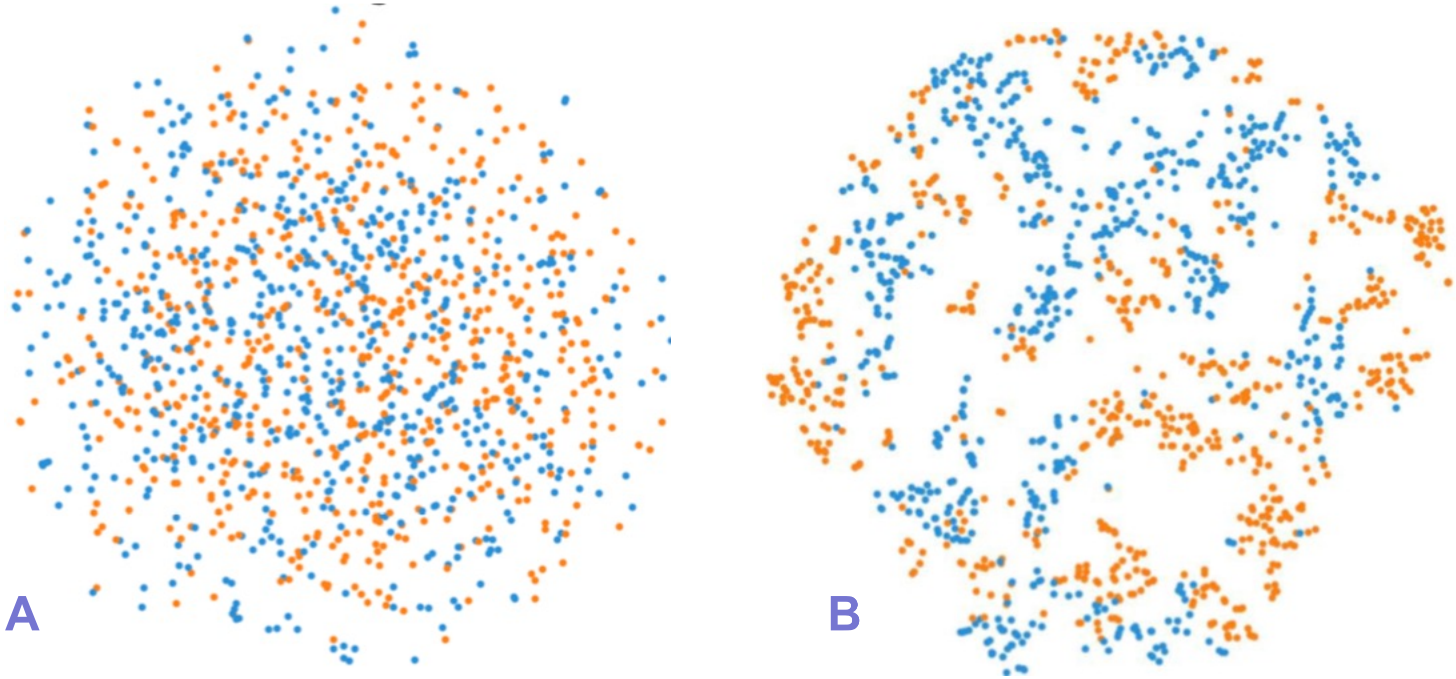
Accuracy

Scalability

Stability

DR for ML

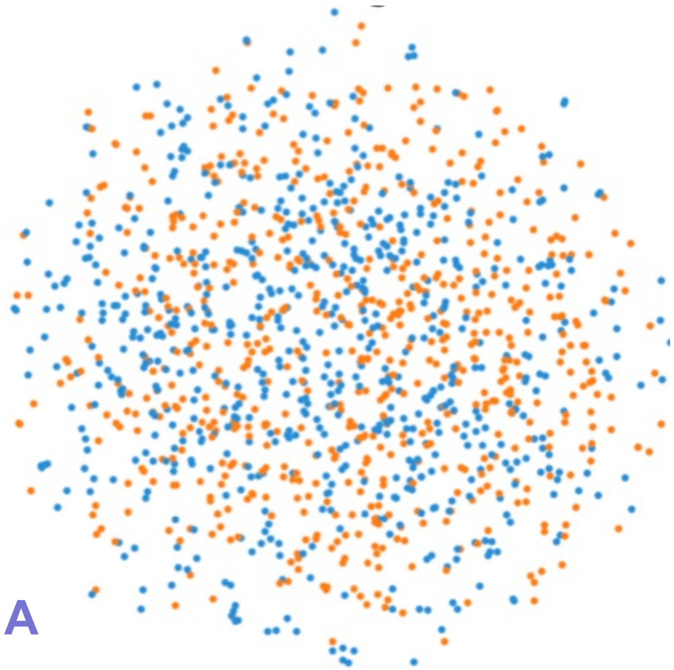
Big Key Idea



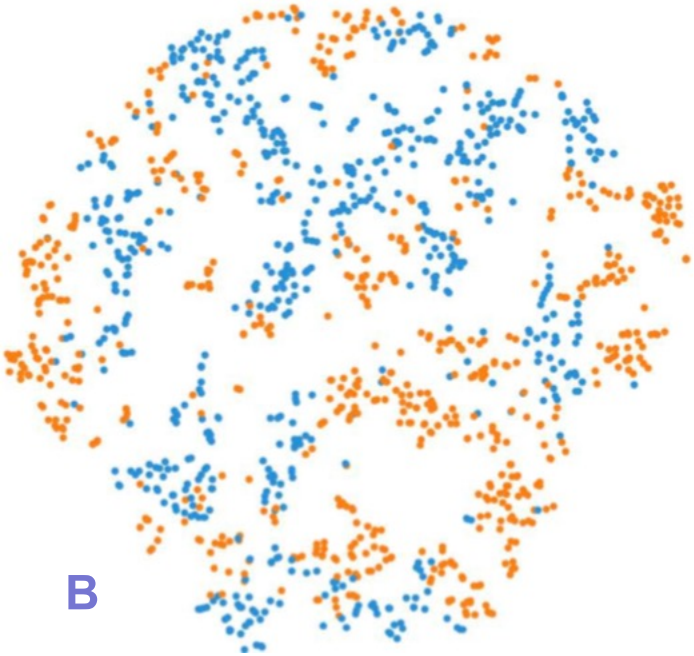
- take two nD feature 2-class datasets A and B
- project them both to 2D using the same DR method
- you see the scatterplots above

Which dataset is easier to classify? Why?

Big Key Idea



A



B

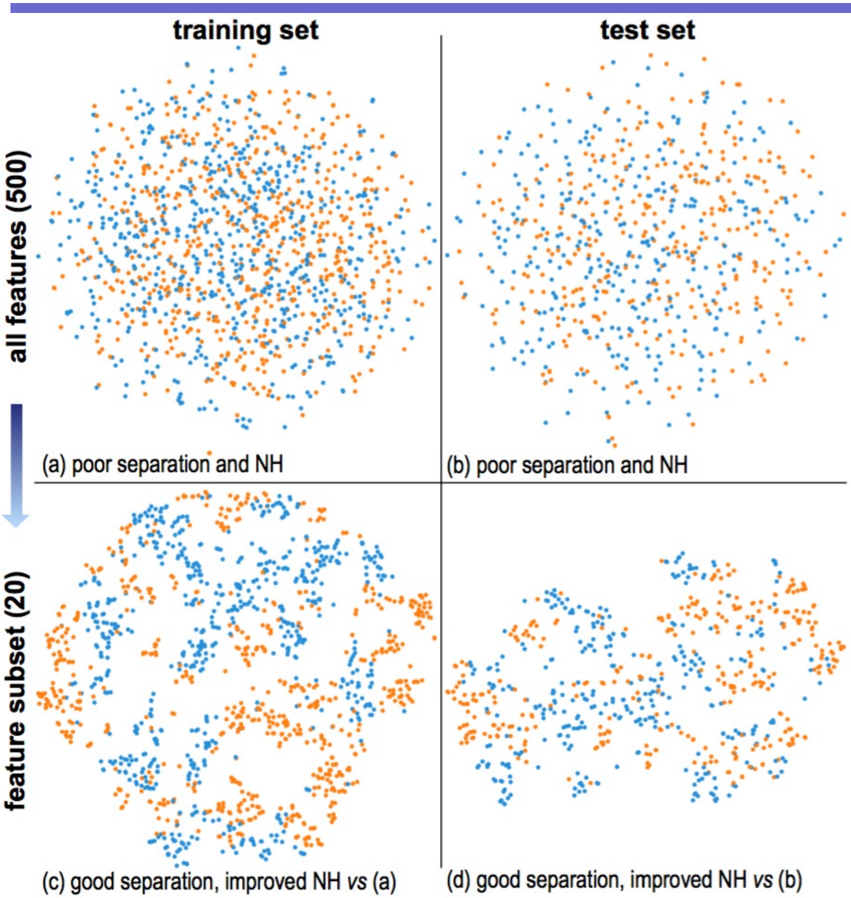
You likely say B is easier. Why?

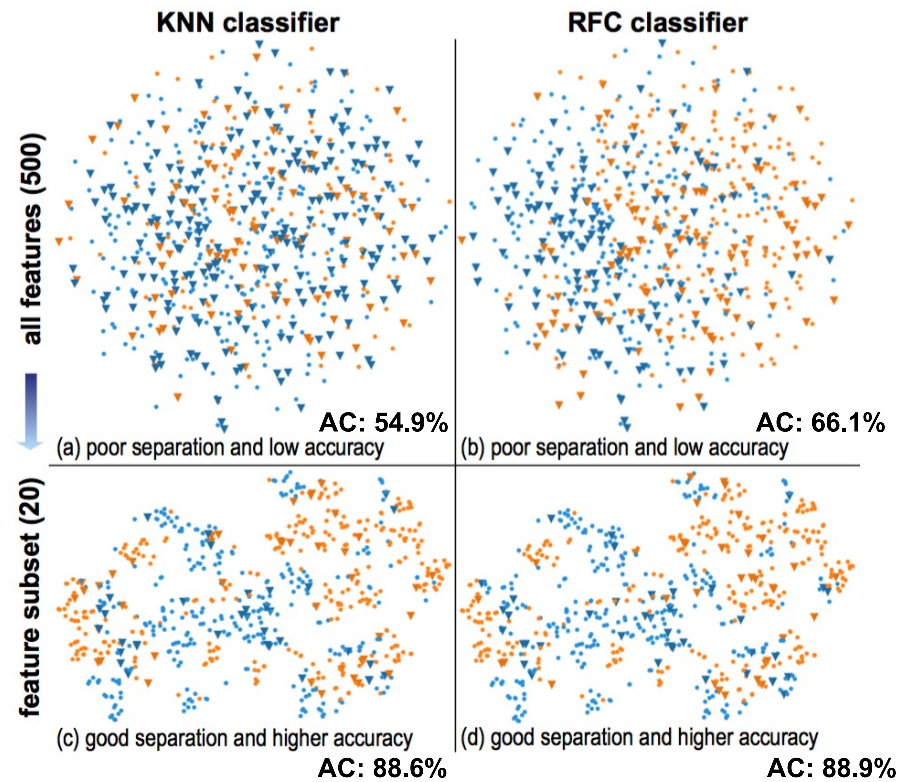
Can we bring evidence for this?

Projections predict classification efficacy

Extensive set of experiments proved that separation in a (good) projection *predicts* accuracy of a (good) classifier

Top row: Hard classification problem → **Validation: many misclassifications**





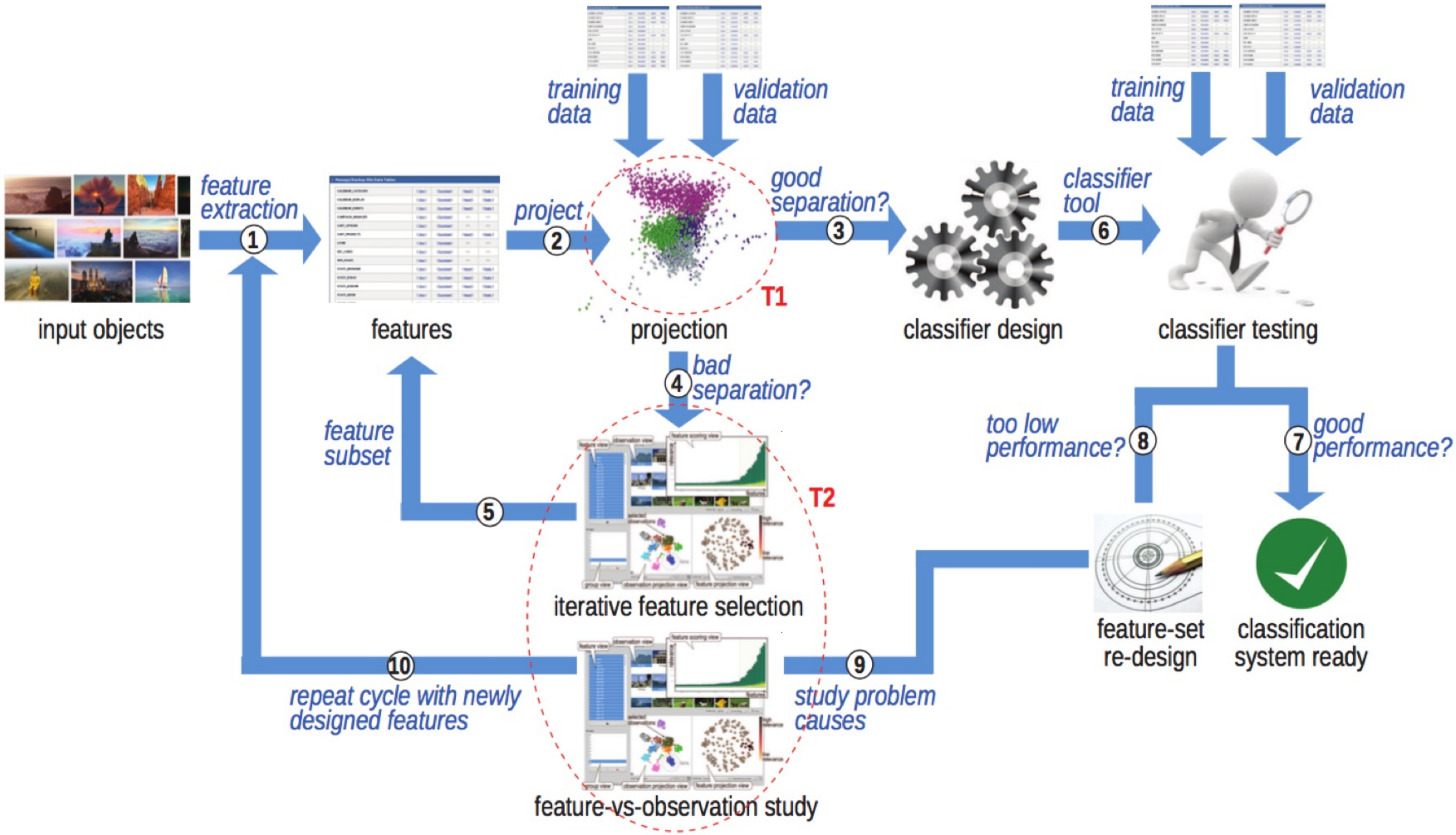
Bottom row: Easy classification (after feature selection) → **Validation: few misclassifications**

Bottom row: Select 20 of 550 features on their discriminative power on training set, using extremely randomized trees

P. Rauber *et al.* (2017) Projections as Visual Aids for Classification System Design. Inf Vis 17(4), 282-305

Projections: Central tool in classifier design

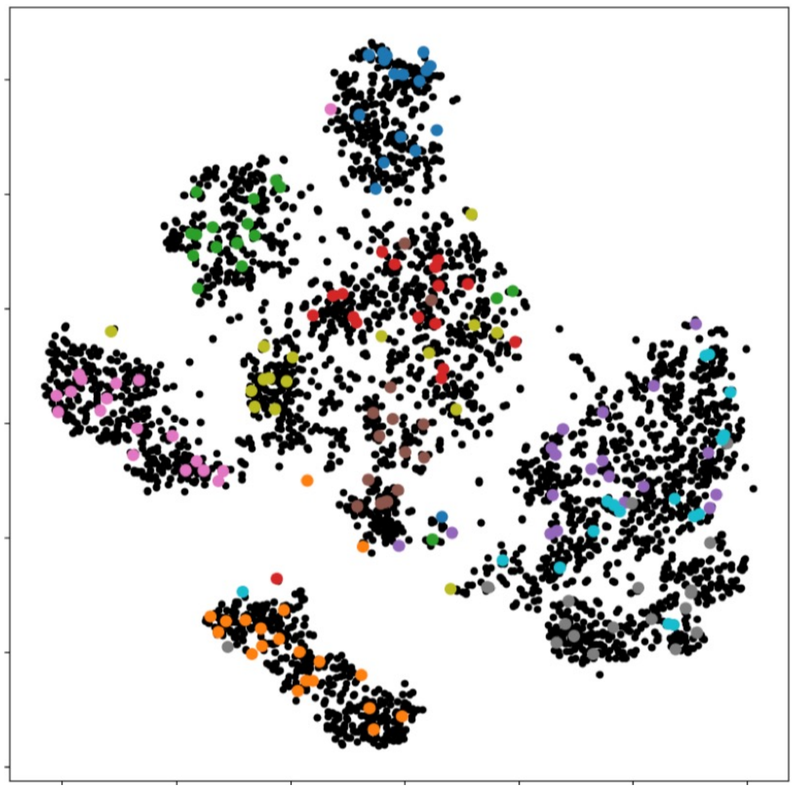
Workflow for two *tasks*



T1: Predict classification efficacy
T2: Improve classification efficacy

Visual Active Learning

We have *under 5%* labeled samples
How to train a classifier with such data?



parasites dataset
colors = manually assigned labels
black = unlabeled samples

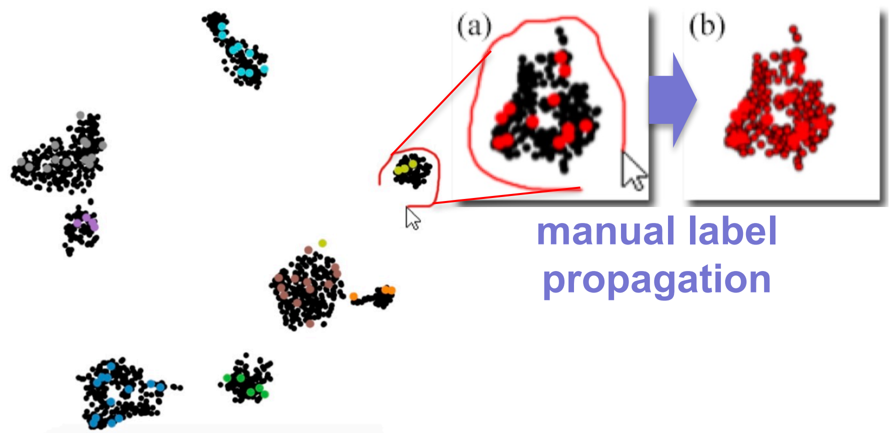
Benato *et* al. (2019) Semi-Supervised Learning with Interactive Label Propagation guided by Feature Space Projections (Proc. SIBGRAPI)

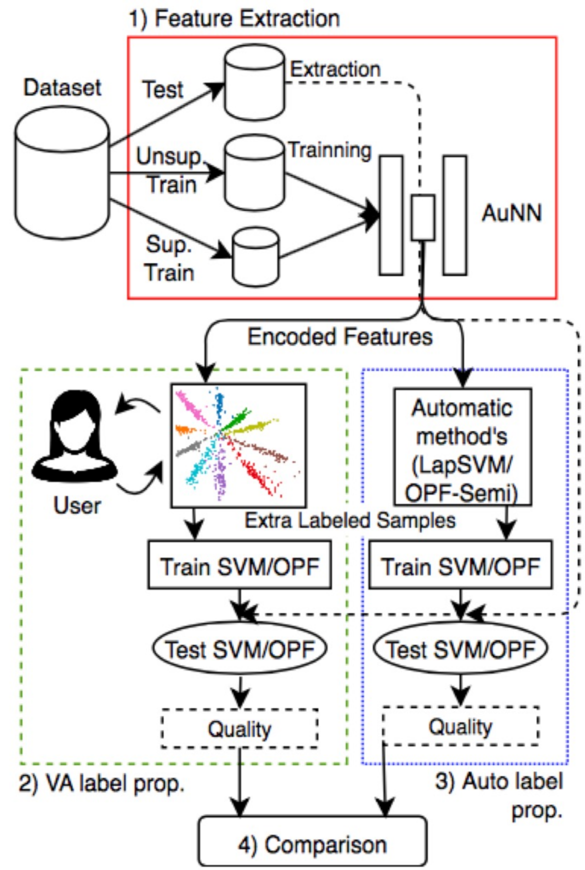
Visual Active Learning: Bringing in the User

Hypotheses

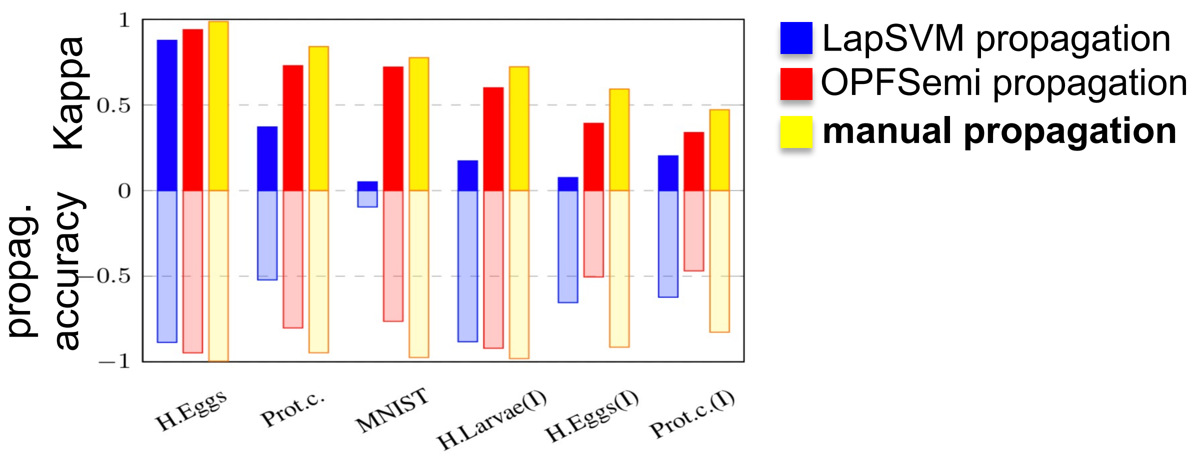
- **similar-label** samples project **close** to each other
- decision boundaries are located in 'empty' area between clusters having different labels

Let's test this!

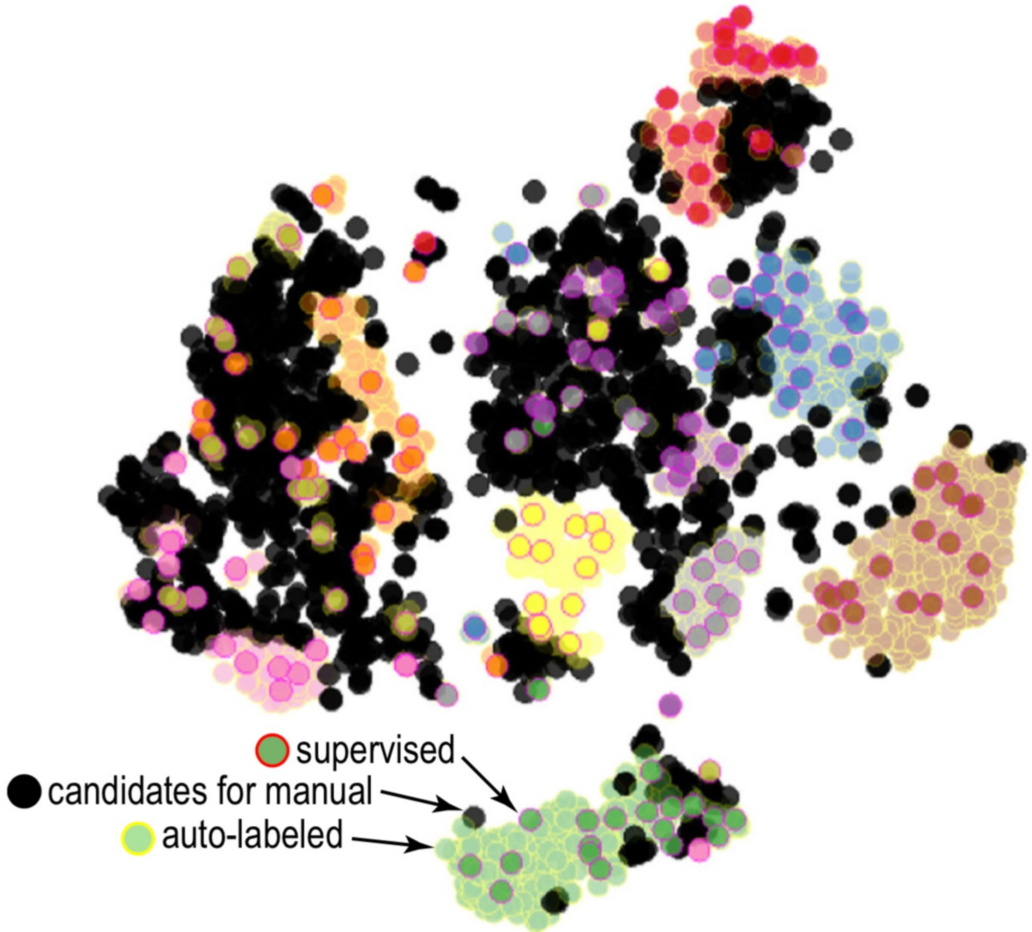




Results



Visual Active Learning: User and Machine Cooperate



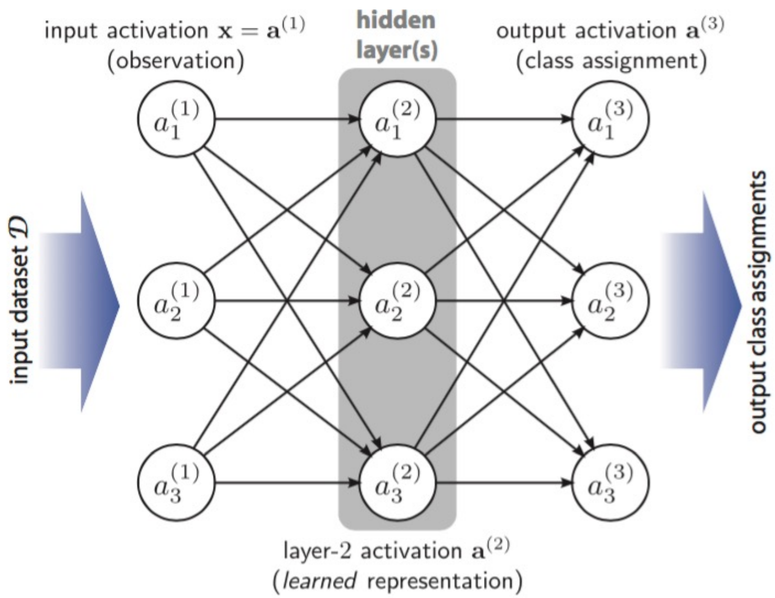
- **automatically** label points where label-propagation confidence is high
- let **user** label (or not) low-confidence points

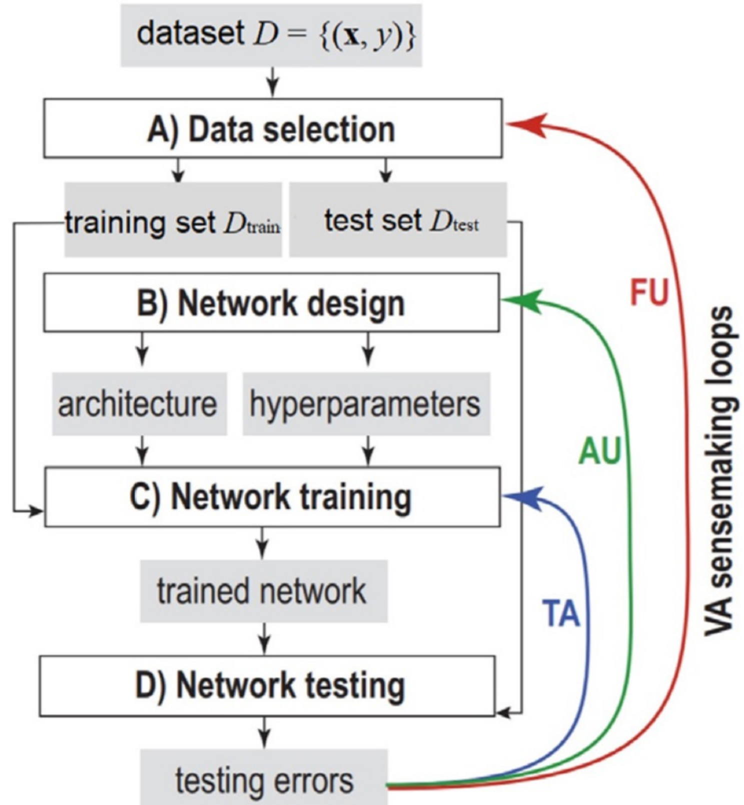
Propagation in 2D space is better than in nD!

Benato *et al.* (2020) Semi-automatic data annotation guided by feature space projection. *Patt Recogn* 109

Projections for understanding deep neural networks

Artificial Neural Networks (ANNs)





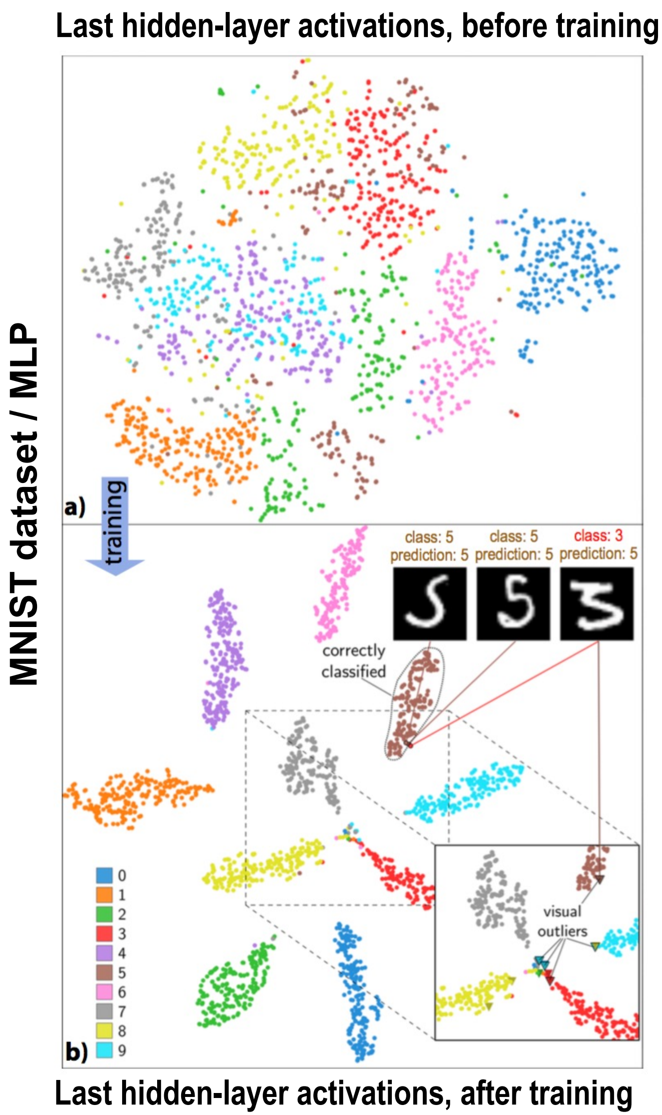
Challenges

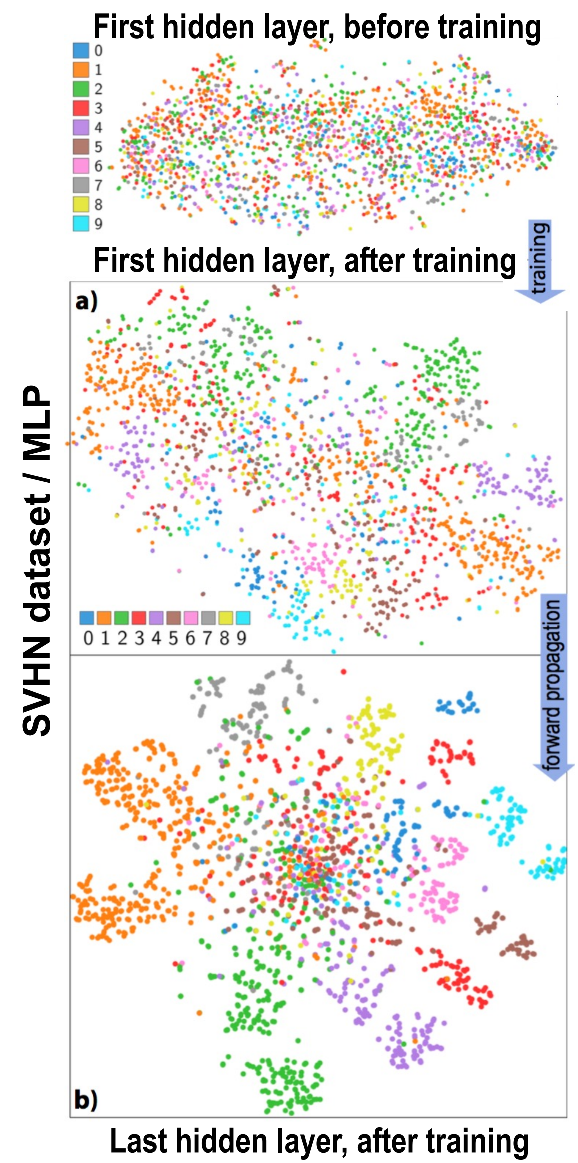
- ANN is a true ‘black box’
- when results are not optimal, how to
    - understand **what** has gone wrong, and **where**?
    - **improve** the classifier?

Explore learned representations (activations)

Method

- project input samples (images) having all activations in a layer as dimensions
- we see how **training** and the **layer** structure create information





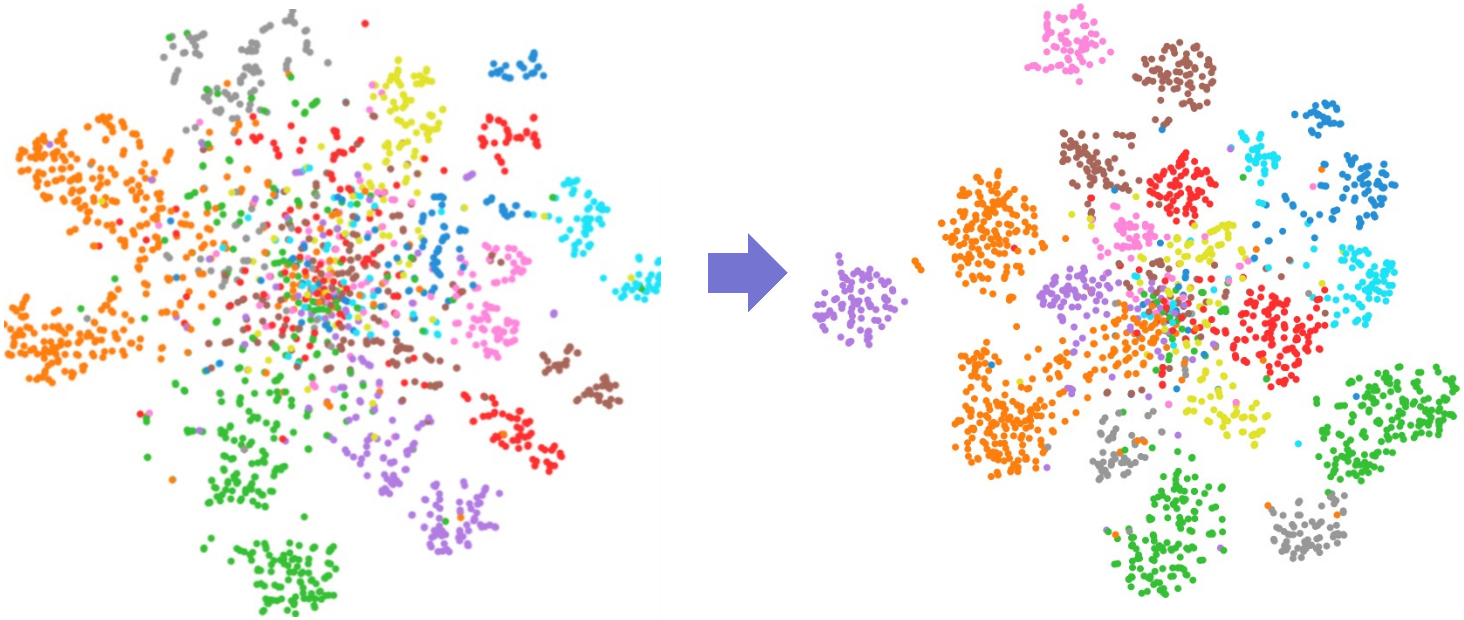
Explore learned representations to improve classification

Example: Compare two architectures (SVHN dataset)

We next notice something strange in this image. Can you see what?

MLP network

CNN network



- reasonable visual separation
  - AC: 77.3%

- much better visual separation
  - AC: 93.8%

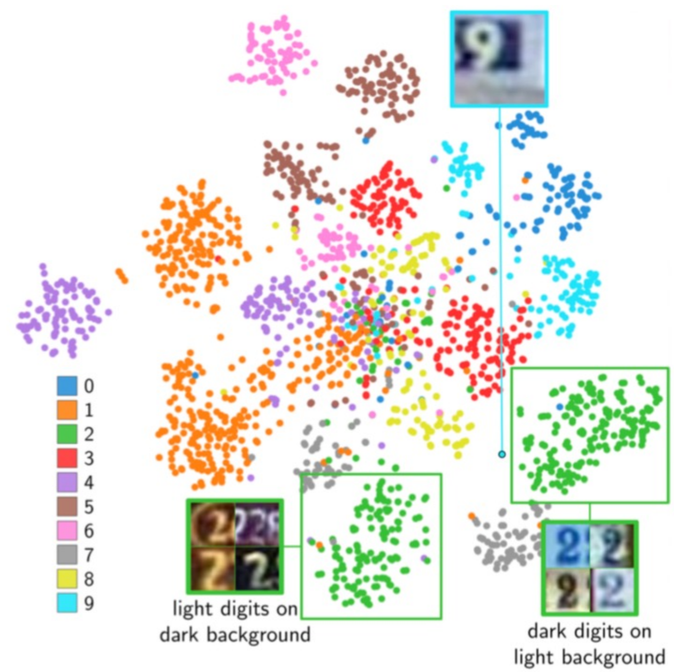
Explore learned representations to improve classification

Each class is formed by two balanced but **distinct clusters!**

MLP network

- reasonable visual separation
  - AC: 77.3%

CNN network



- much better visual separation
  - AC: 93.8%

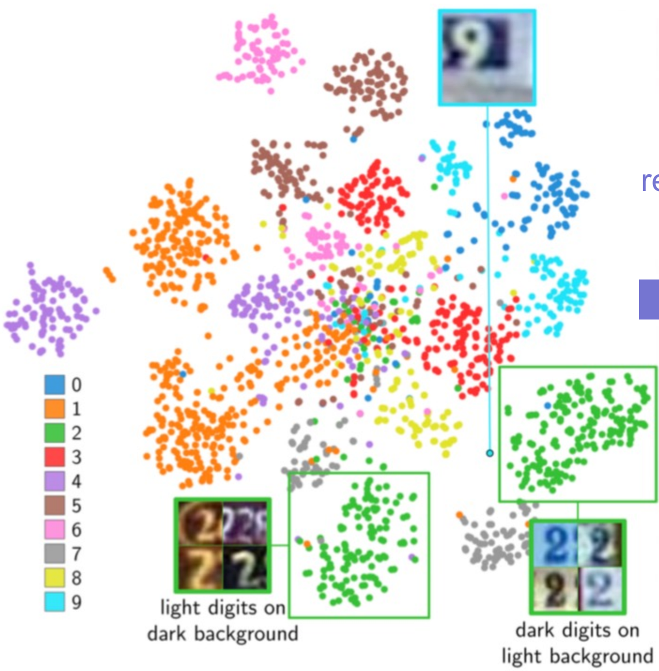
Explore learned representations to improve classification

What is going on?

- visually explore clusters by brushing
- we find that each cluster-pair contains
  - a cluster for light images on **dark** background
  - a cluster for **dark** images on light background

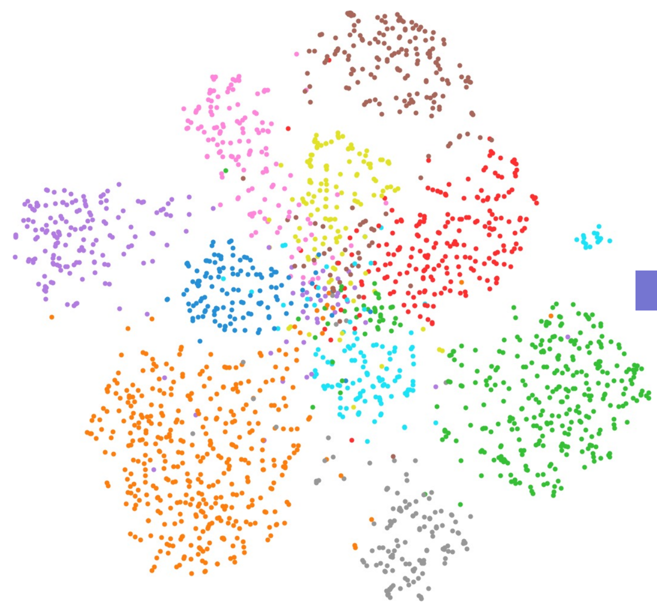
Let's use this insight to improve the classification

Current situation



replace images by edge-detection versions

After preprocessing



Accuracy increases

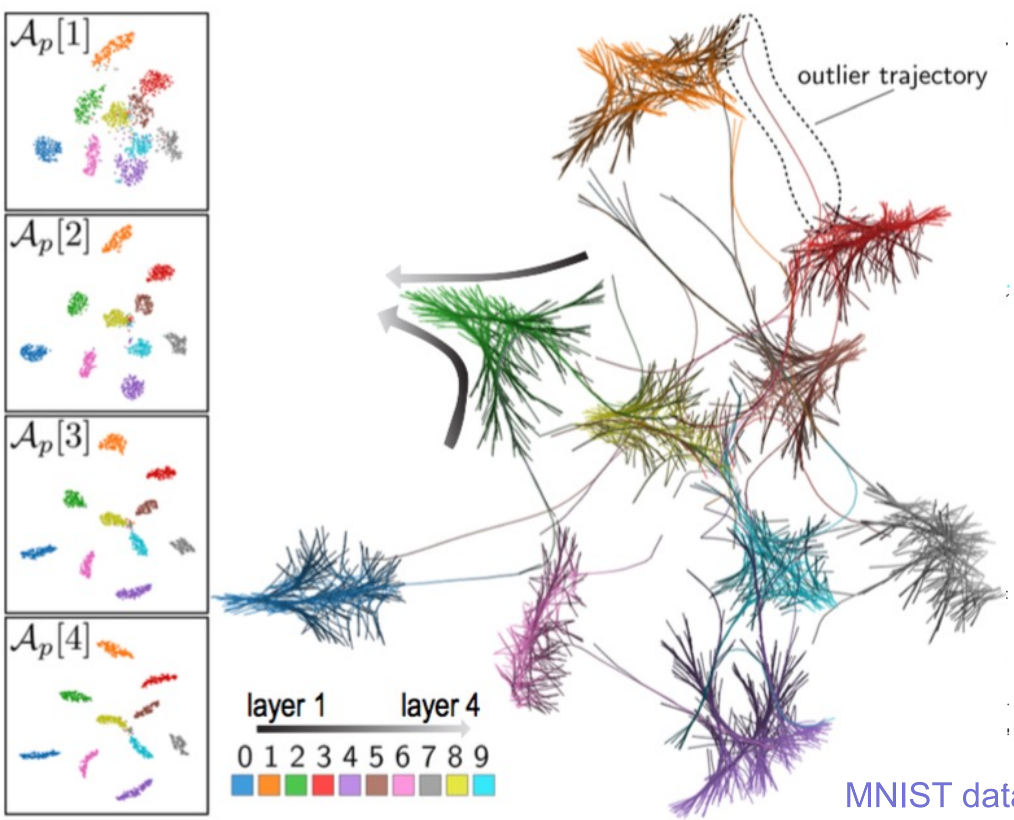
MLP: +4%
CNN: +0.7%

Explore evolution of learned representations

Activation in an ANN change in time in two ways
- as data flows from the 1st to the last layer during operation (**inter-layer** evolution)
- as different datasets are used during training (**inter-epoch** evolution)

We want to explore both to
- understand how different layers contribute to learning
- understand if training is effective

Inter-layer evolution



MNIST dataset, MLP classifier

Bundled observation paths (built using dynamic t-SNE)

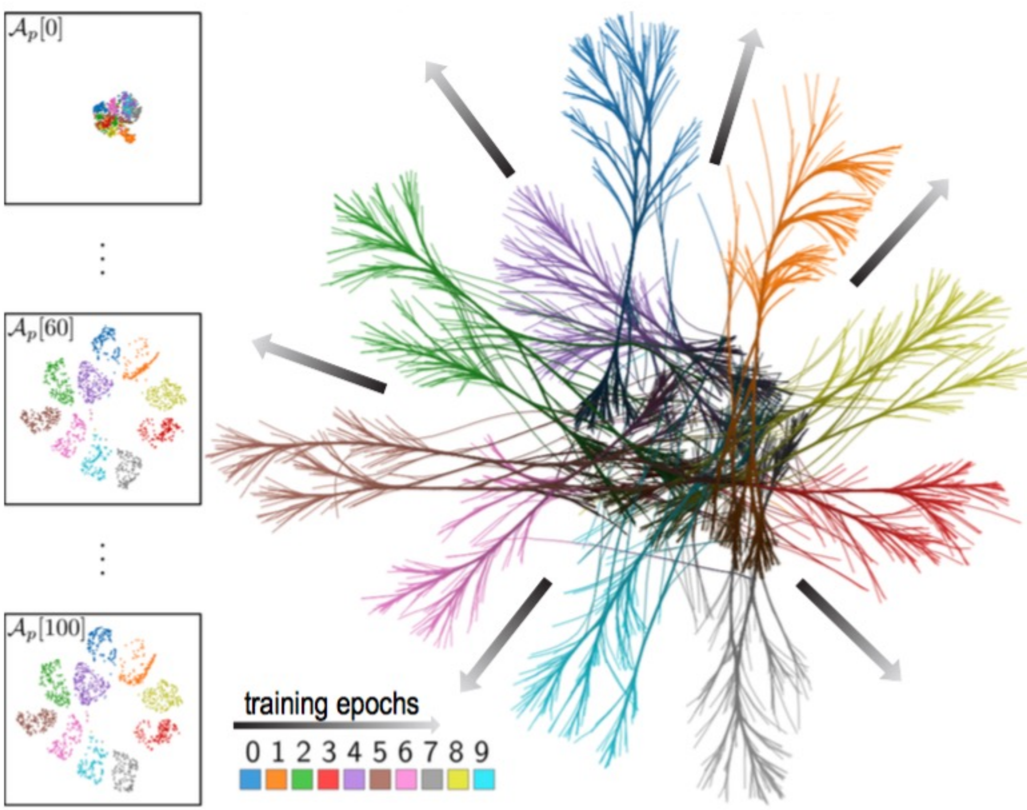
We observe how
- group separation increases
- group size decreases
- groups increasingly diverge
- few trails connect different groups (classification decisions are stable)

Conclusions

Network performs (very) well!

Explore evolution of learned representations

Inter-epoch evolution



MNIST dataset, last CNN hidden layer, 100 training epochs

Bundled observation paths

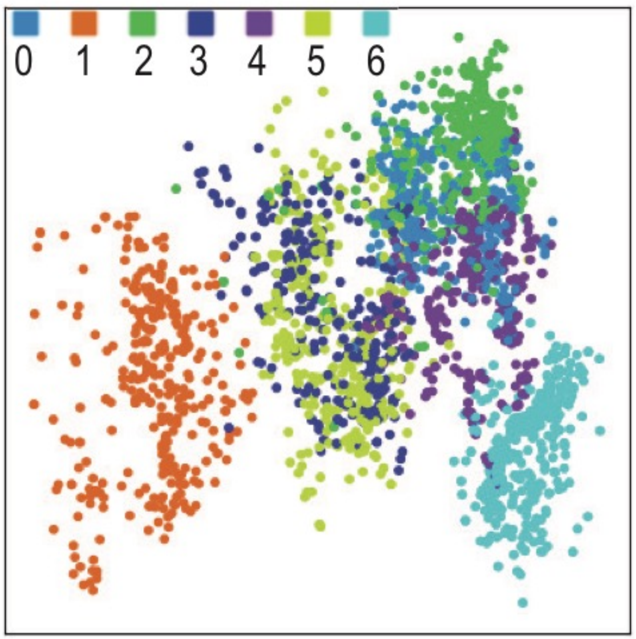
We observe how
- group separation increases (from complete clutter to perfect separation)
- groups increasingly diverge
- paths are quite straight/smooth (no canceling of learning)
- paths don't link different-color groups

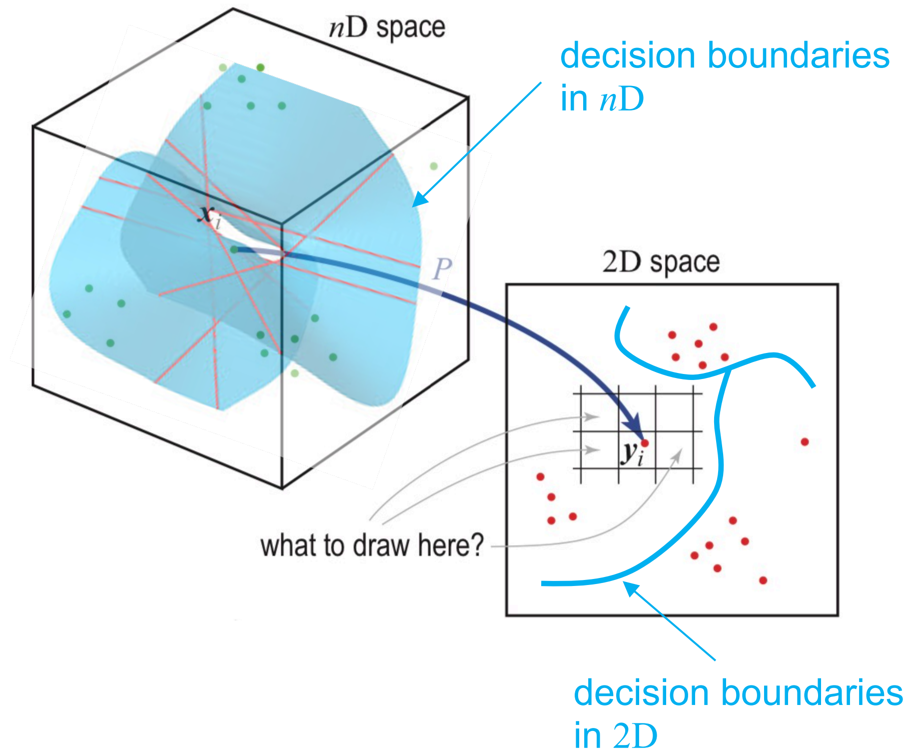
Conclusions

- learning is very effective
- knowledge accumulates as desired
- few/no 'learning hesitations'

Classifier Decision Maps

Often referred to in many ML papers, rarely *shown*

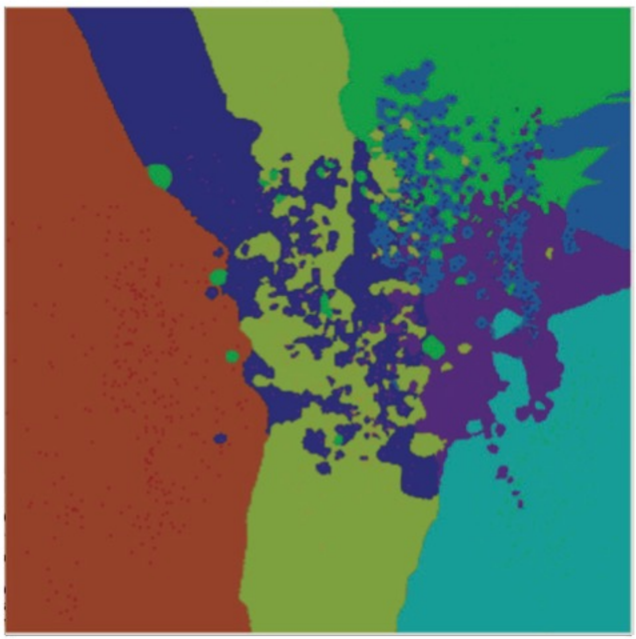


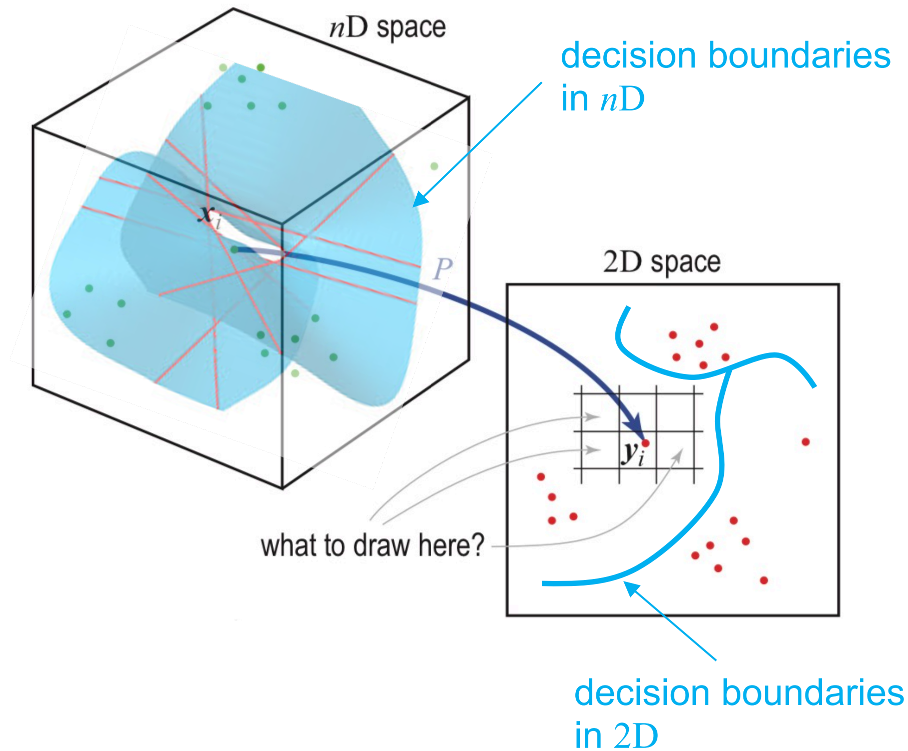


Question: How can we actually *see* those decision boundaries?

Classifier Decision Maps

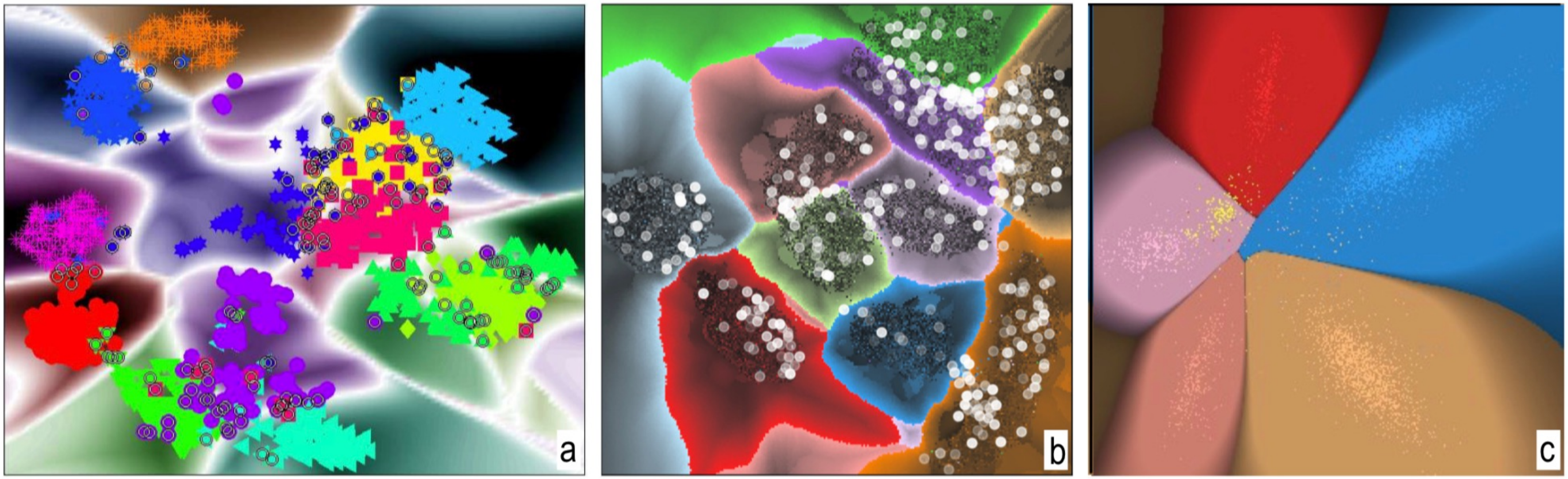
Often referred to in many ML papers, rarely *shown*





Yes, we can visualize decision boundaries using projections!

Classifier Decision Maps



luminance: classifier confidence

luminance: distance to boundary

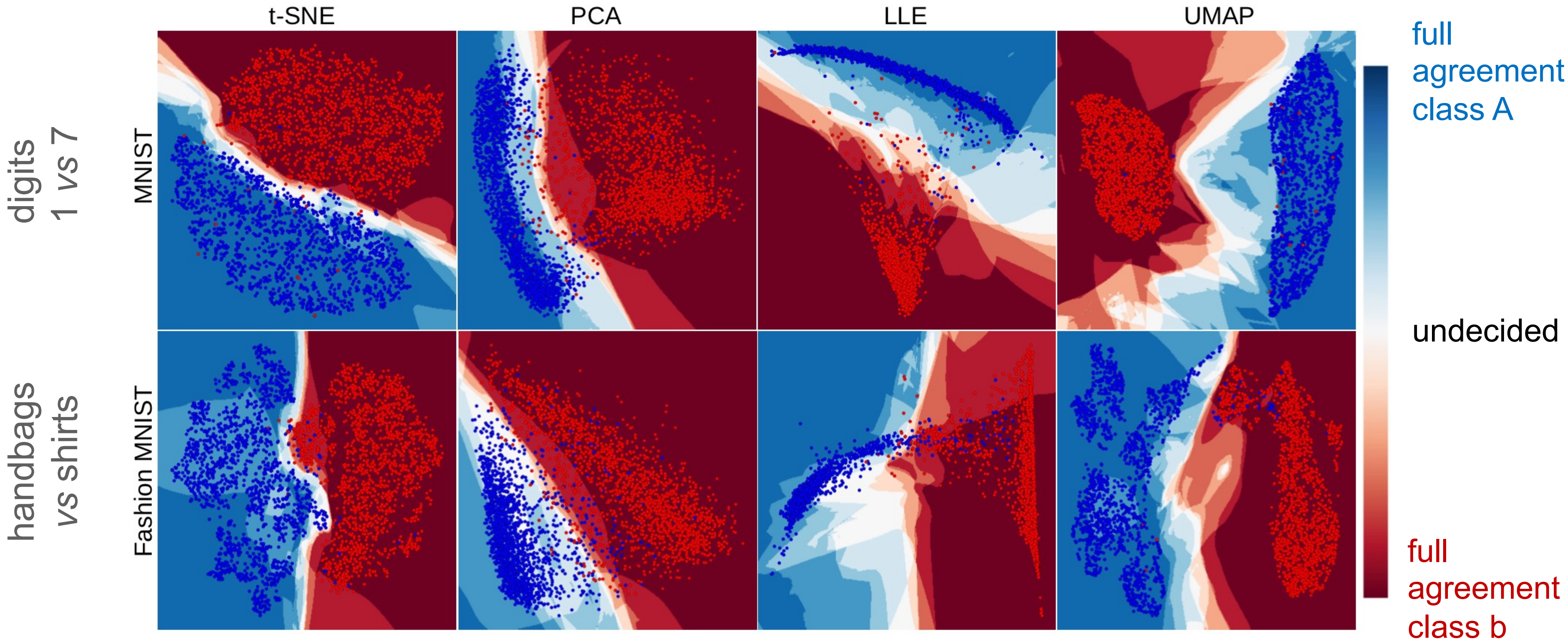
luminance: classifier confidence

Key to solution: Construct *inverse* projection from 2D to nD

Espadoto *et al.* (2021) UnProjection: Leveraging inverse-projections for visual analytics of high-dimensional data. IEEE TVCG
Schulz *et al.* (2015) Using discriminative dimensionality reduction to visualize classifiers. Neural Process Lett 42(1)

Multi-classifier decision maps

Visualize agreement of nine classifiers on two-class problems



Classifiers used

- Logistic regression, Linear SVM (linear), SVM (RBF), kNN, Gaussian Process, Decision Tree, Random Forest, Adaboost, Gaussian Naive Bayes, Quadratic Discriminant Analysis

- t-SNE: small cluster separation but narrow white band (low uncertainty)
- UMAP: large cluster separation but thick white band (high uncertainty)

M. Espadoto *et al.* (2021) UnProjection: Leveraging Inverse-Projections for Visual Analytics of High-Dimensional Data. IEEE TVCG

Which projection technique is the “best”?

Projections are clearly useful tools for ML engineering
But which projection is the *best* to use in practice?

PCA: Poor separation...

Isomap: Bit better separation...

t-SNE: Strong separation

UMAP: Extreme separation

Which DR technique to *use* ?

surveys

techniques

Projection Acronym	Projection Full Name	Fodor *et al.* [18]	Hoffman *et al.* [1]	Yin *et al.* [19]	Maaten *et al.* [13]	Bunte *et al.* [15]	Engel *et al.* [27]	Sorzano *et al.* [12]	Cunningham *et al.* [23]	Gisbrecht *et al.* [21]	Liu *et al.* [2]	Xie *et al.* [24]	Nonato *et al.* [10]	**Ours**
AE	Autoencoder				•									•
CCA	CCA (Canonical Correlations Analysis)								•					
CHL	Chalmers												•	
CLM	ClassiMap												•	
CuCA	CCA (Curvilinear Component Analysis)												•	
DM	Diffusion Maps				•									•
DML	Distance Metric Learning								•					
EM	Elastic Maps							•						
FA	Factor Analysis	•						•	•					•
FD	Force-Directed												•	
FMAP	FastMap												•	•
FS	Feature Selection											•		
GDA	Generalized Discriminant Analysis													•
GPLVM	Gaussian Process Latent Variable Model													•
GTM	Generative Topographic Mapping							•					•	
ICA	Independent Component Analysis	•						•	•					
F-ICA	FastICA													•
NL-ICA	Nonlinear ICA	•												
IDMAP	IDMAP													•
ISO	Isomap		•	•	•	•	•			•			•	•
L-ISO	Landmark Isomap													•
KECA	Kernel Entropy Component Analysis							•						
KLP	Kelp												•	
LAMP	LAMP												•	•
LDA	Linear Discriminant Analysis								•		•	•	•	
LE	Laplacian Eigenmaps				•	•				•	•		•	•
LLC	Locally Linear Coordination				•									•
LLE	Locally Linear Embedding		•	•	•	•	•			•	•		•	•
H-LLE	Hessian LLE				•									•
M-LLE	Modified LLE													•
LMNN	Large-Margin Nearest Neighbor Metric													•
LoCH	Local Convex Hull												•	
LPP	Locality Preserving Projection								•					•
LR	Linear Regression								•					
LSP	Least Square Projection												•	•
LTSA	Local Tangent Space Alignment				•								•	•
L-LTSA	Linear Local Tangent Space Alignment													•
MAF	Maximum Autocorrelation Factors								•					
MC	Manifold Charting				•					•				•
MCA	Multiple Correspondence Analysis												•	
MCML	Maximally Collapsing Metric Learning													•
MDS	Metric Multidimensional Scaling	•	•	•	•	•	•	•	•		•		•	•
L-MDS	Landmark MDS													•
MG-MDS	Multi-Grid MDS						•							
N-MDS	Nonmetric MDS (Kruskal)		•				•						•	•
ML	Manifold Learning						•							
MVU	Maximum Variance Unfolding				•	•				•			•	
FMVU	Fast MVU													
L-MVU	Landmark MVU													•
NeRV	Neighborhood Retrieval Visualizer					•								
t-NeRV	t-NeRV					•								
NMF	Nonnegative Matrix Factorization							•	•					•
NLM	Nonlinear Mapping												•	
NN	Neural Networks	•												
PBC	Projection By Clustering													•
PC	Principal Curves	•						•						
PCA	Principal Component Analysis	•	•		•		•	•	•	•	•	•	•	•
I-PCA	Incremental PCA							•						•
K-PCA-P	Kernel PCA (Polynomial)													•
K-PCA-R	Kernel PCA (RBF)		•		•		•	•		•				•
K-PCA-S	Kernel PCA (Sigmoid)													•
L-PCA	Localized PCA							•						
NL-PCA	Nonlinear PCA	•		•				•						
P-PCA	Probabilistic PCA								•					•
R-PCA	Robust PCA							•						
S-PCA	Sparse PCA							•						•
PLMP	Part-Linear Multidimensional Projection												•	
PLP	Piecewise Laplacian-based Projection						•						•	
PLSP	Piecewise Least Square Projection													•
PM	Principal Manifolds			•										
PP	Projection Pursuit	•												
RBF-MP	RBF Multidimensional Projection												•	
RP	Random Projections	•										•		
G-RP	Gaussian Random Projection													•
S-RP	Sparse Random Projection													•
SAM	Sammon Mapping				•									
R-SAM	Rapid Sammon (Pekalska)												•	•
SDR	Sufficient Dimensionality Reduction								•					
SFA	Slow Feature Analysis								•					
SMA	Smacof												•	
SNE	Stochastic Neighborhood Embedding					•							•	
T-SNE	t-Dist. Stochastic Neighborhood Embedding					•				•	•			•
SOM	Self-Organizing Maps	•		•				•					•	
ViSOM	ViSOM (Visualization-induced SOM)			•										
SPE	Stochastic Proximity Embedding													•
G-SVD	Generalized SVD							•						
T-SVD	Truncated SVD													•
TF	Tensor Factorization							•						
UMAP	Uniform Manifold Approximation and Proj.													•
VQ	Vector Quantization	•						•						
Total		12	6	7	14	9	9	19	14	8	6	4	28	44

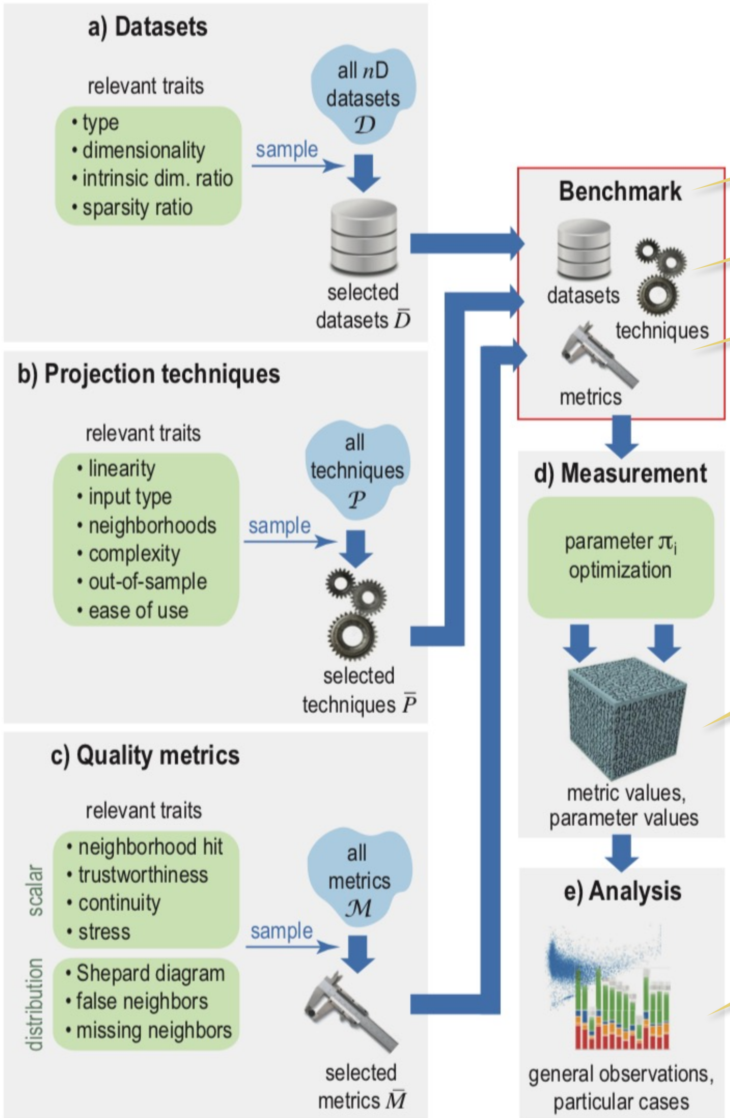
Big and unclear 'choice space'

- 50+ techniques
- 12 main surveys
- mainly theoretical discussion
- many parameters
- very limited practical comparison

Practitioner questions

- which projection is **best** for **my** context (requirements, data, …)?
- how to set its **parameters**?
- how to measure its **quality**?

Let's measure projection errors big-scale!



19 datasets

45 techniques

6 metrics

parameter analysis

insights

M. Espadoto *et al* (2019) Towards a Quantitative Survey of Dimension Reduction Techniques (IEEE TVCG)

Datasets and Metrics

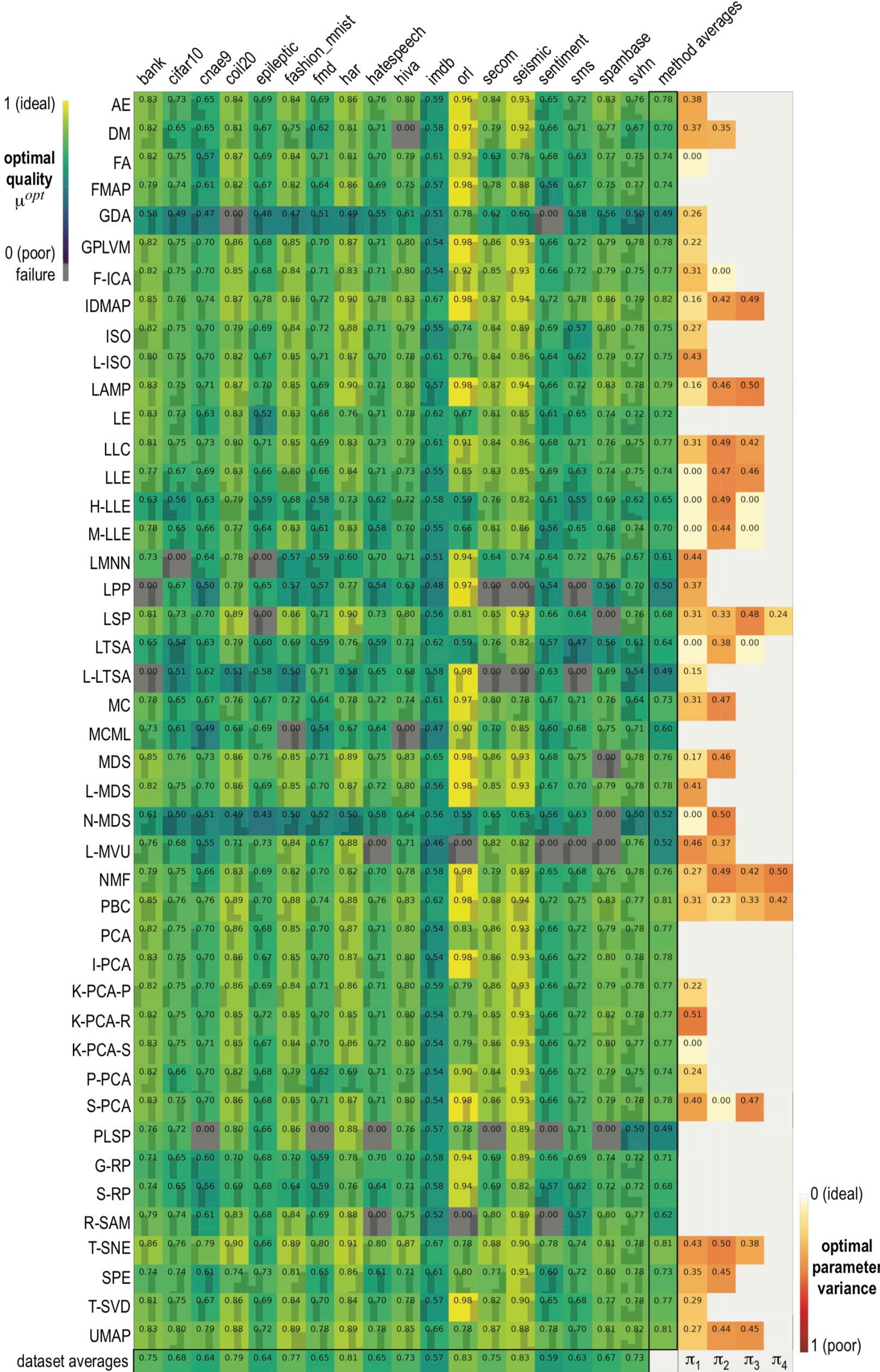
Datasets

Dataset	Type (τ_D)	Size (N)	Size class	Dimensionality (n)	Dimensionality class	Intrinsic dim. (ρ_n)	Intrinsic dim. class	Sparsity (γ_n)	Sparsity class
bank	*tables*	2059	*medium*	63	*low*	0.0317	*low*	0.6963	*medium*
cifar10	*images*	3250	*large*	1024	*high*	0.0706	*low*	0.0024	*dense*
cnae9	*text*	1080	*medium*	856	*high*	0.3201	*medium*	0.9922	*sparse*
coil20	*images*	1440	*medium*	400	*medium*	0.0105	*low*	0.3858	*medium*
epileptic	*tables*	5750	*large*	178	*medium*	0.2191	*medium*	0.0067	*dense*
fashion_mnist	*images*	3000	*medium*	784	*high*	0.2385	*medium*	0.5021	*medium*
fmd	*images*	997	*small*	1536	*high*	0.3073	*medium*	0.0095	*dense*
har	*tables*	735	*small*	561	*high*	0.1194	*medium*	0.0001	*dense*
hatespeech	*text*	3222	*large*	100	*medium*	0.6130	*high*	0.9993	*sparse*
hiva	*tables*	3076	*large*	1617	*high*	0.2498	*medium*	0.9091	*sparse*
imdb	*text*	3250	*large*	700	*high*	0.5790	*high*	0.9945	*sparse*
orl	*images*	400	*small*	396	*medium*	0.0006	*low*	0.9000	*sparse*
secom	*tables*	1567	*medium*	590	*high*	0.0102	*low*	0.2617	*medium*
seismic	*tables*	646	*small*	24	*low*	0.0417	*low*	0.5883	*medium*
sentiment	*text*	2748	*medium*	200	*medium*	0.8080	*high*	0.9936	*sparse*
sms	*text*	836	*small*	500	*medium*	0.7240	*high*	0.9947	*sparse*
spambase	*text*	4601	*large*	57	*low*	0.0351	*low*	0.7741	*medium*
svhn	*images*	733	*small*	1024	*high*	0.8734	*high*	0.0001	*dense*

Metrics

Metric	**Definition**	**Type**	**Range**
Trustworthiness (M_t)	$1 - \frac{2}{NK(2n-3K-1)} \sum_{i=1}^{N} \sum_{j \in U_i^{(K)}} (r(i,j) - K)$	scalar	$[0, \mathbf{1}]$
Continuity (M_c)	$1 - \frac{2}{NK(2n-3K-1)} \sum_{i=1}^{N} \sum_{j \in V_i^{(K)}} (\hat{r}(i,j) - K)$	scalar	$[0, \mathbf{1}]$
Normalized stress (M_σ)	$\frac{\sum_{ij} (\Delta^n(\mathbf{x}_i, \mathbf{x}_j) - \Delta^q(P(\mathbf{x}_i), P(\mathbf{x}_j)))^2}{\sum_{ij} \Delta^n(\mathbf{x}_i, \mathbf{x}_j)^2}$	scalar	$[\mathbf{0}, 1]$
Neighborhood hit (M_{NH})	$\sum_{i=1}^{N} \frac{\lvert j \in N_i^{(K)} : l_j = l_i \rvert}{KN}$	scalar	$[0, \mathbf{1}]$
Shepard diagram (S)	Scatterplot $(\lVert \mathbf{x}_i - \mathbf{x}_j \rVert, \lVert P(\mathbf{x}_i) - P(\mathbf{x}_j) \rVert), 1 \leq i \leq N, i \neq j$	point-pair	-
Shepard goodness (M_S)	Spearman rank correlation of Shepard diagram	scalar	$[0, \mathbf{1}]$
Average local error ($M_a(i)$)	$\frac{1}{N-1} \sum_{j \neq i} \left\lvert \frac{\Delta^n(\mathbf{x}_i, \mathbf{x}_j)}{\max_{i,j} \Delta^n(\mathbf{x}_i, \mathbf{x}_j)} - \frac{\Delta^q(P(\mathbf{x}_i), P(\mathbf{x}_j))}{\max_{i,j} \Delta^q(P(\mathbf{x}_i), P(\mathbf{x}_j))} \right\rvert$	local (per-point)	$[\mathbf{0}, 1]$

aggregate into a single quality metric μ



Insights (1)

How good are projections, for which data?

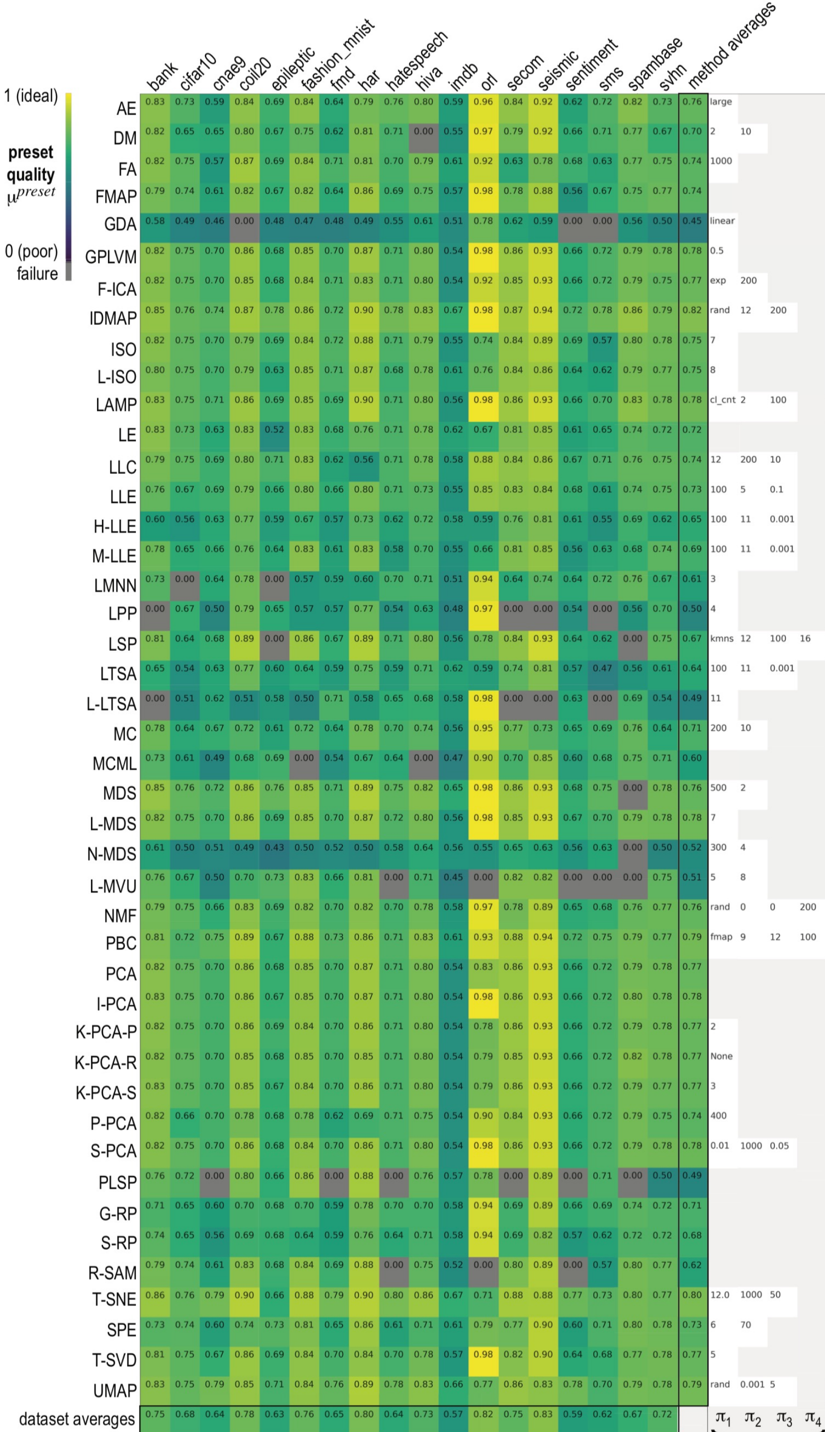
for each projection P_i
for each dataset D_j
compute *optimal* quality μ_{ij} (param. grid search)

How easy is to get optimal quality?

for each projection P_i
compute *variance* of params π_i yielding optimal quality over all datasets D_j

What we see

- no projection **best** for all dataset types
- some are quite **poor** in general (N-MDS, GDA)
- dataset **type** strongly influences quality (*imdb*: hard; *orl*: easy)
- hard to **tune** parameters to get optimal quality (large variance of π_i)



Insights (2)

How good are parameter-preset projections?

for each projection P_i

π_i^{pre} = param values yielding most times optimal quality over all datasets D_j

for each projection P_i

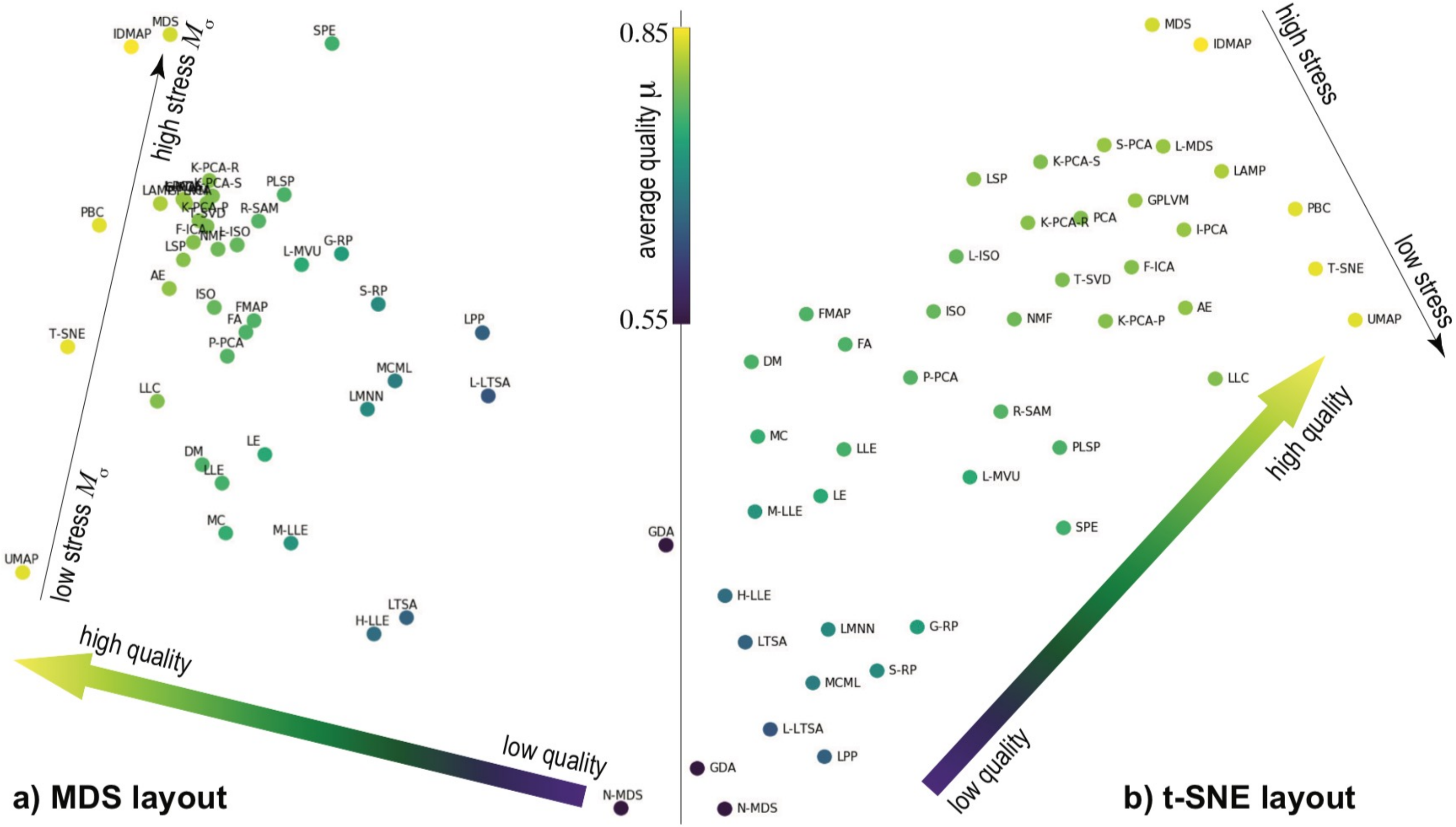
for each dataset D_j

compute quality μ_{ij} using π_i^{pre}

What we see

- very similar image to earlier one (optimal techniques stay good when using **presets**)
- again, quality strongly depends on dataset type
- t-SNE, UMAP, IDMAP, PBC score **best** on average

Insights (3): Which projections perform similarly?



‘Projection of projections’ map

- one point = one technique
- 5 attributes (trustworthiness, continuity, norm. stress, neighborhood hit, Shepard goodness; averaged over all tested datasets)
- we see a clear **quality trend**
- helps choosing projections that behave similarly to a user-chosen one

Benchmark

Towards A Quantitative Survey of Dimension Reduction Techniques

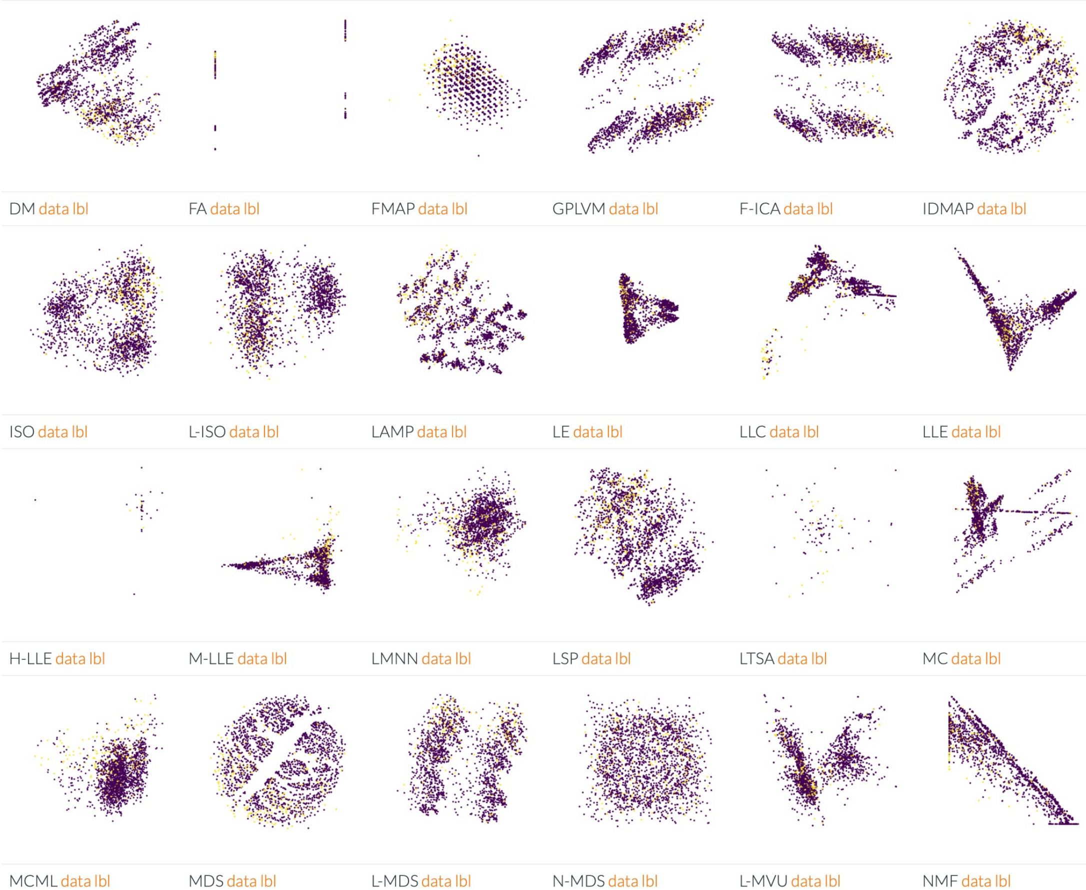
MATEUS ESPADOTO, RAFAEL M. MARTINS, ANDREAS KERREN, NINA S. T. HIRATA AND ALEXANDRU C. TELEA

DATASETS EXPERIMENT MEASUREMENTS PROJECTIONS

Projections for all datasets (best parameter set for each projection)

All projections, in csv format

bank dataset

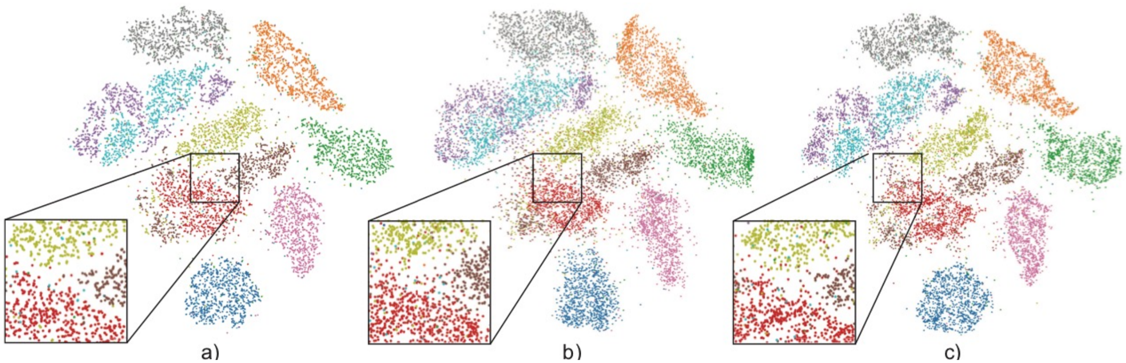


https://mespadoto.github.io/proj-quant-eval

All open source

- projection implementations
- datasets
- metric engines
- visualization engines
- optimization engines
- test harness
- all Python code

Please share, use, and extend!



ML for DR

Insights from our DR survey

No ideal projection technique ☹

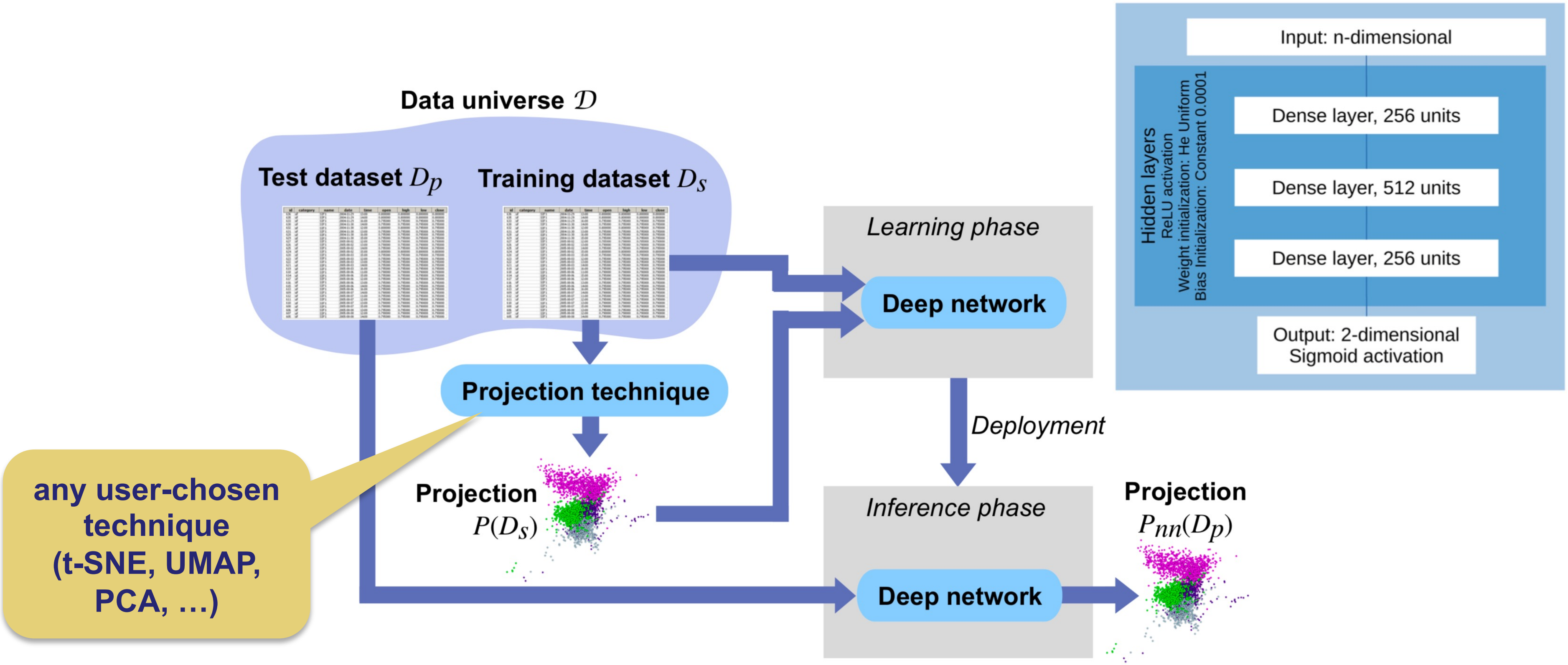
- UMAP: easy to use, quite fast, but quality not ideal
- t-SNE: quality is (very) high, but very slow, hard to tweak parameters, non-deterministic
- quality depends a lot on type of data

What we want to ultimately have

- high-**quality** projection
- having `**style**' of any projection deemed good by user
- working very **fast** (millions of samples, hundreds of dimensions: seconds)
- **easy** to use (no complex parameters, ideally none)
- **stable** (same input data: same output projection)
- **out-of-sample** (add some more data: project along existing data)

How to achieve this?

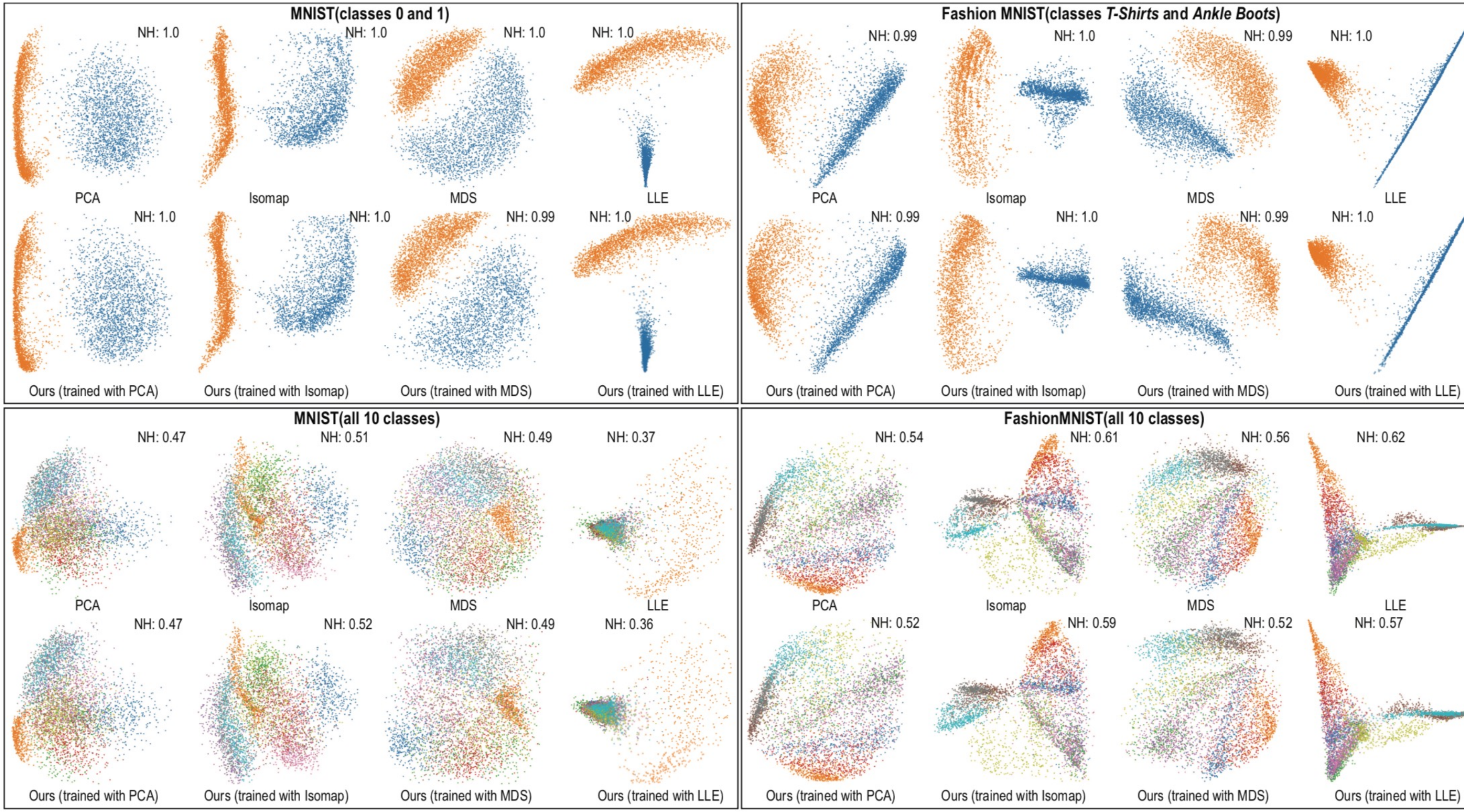
Idea: Learn the projection!



- take *any* dataset D_S and *any* projection technique of choice P
- project D_S with P, tweak P's parameters, obtain good scatterplot $P(D_S)$
- pass D_S and $P(D_S)$ to network, learn the mapping
- use trained network P_{nn} to project any other similar dataset D_P

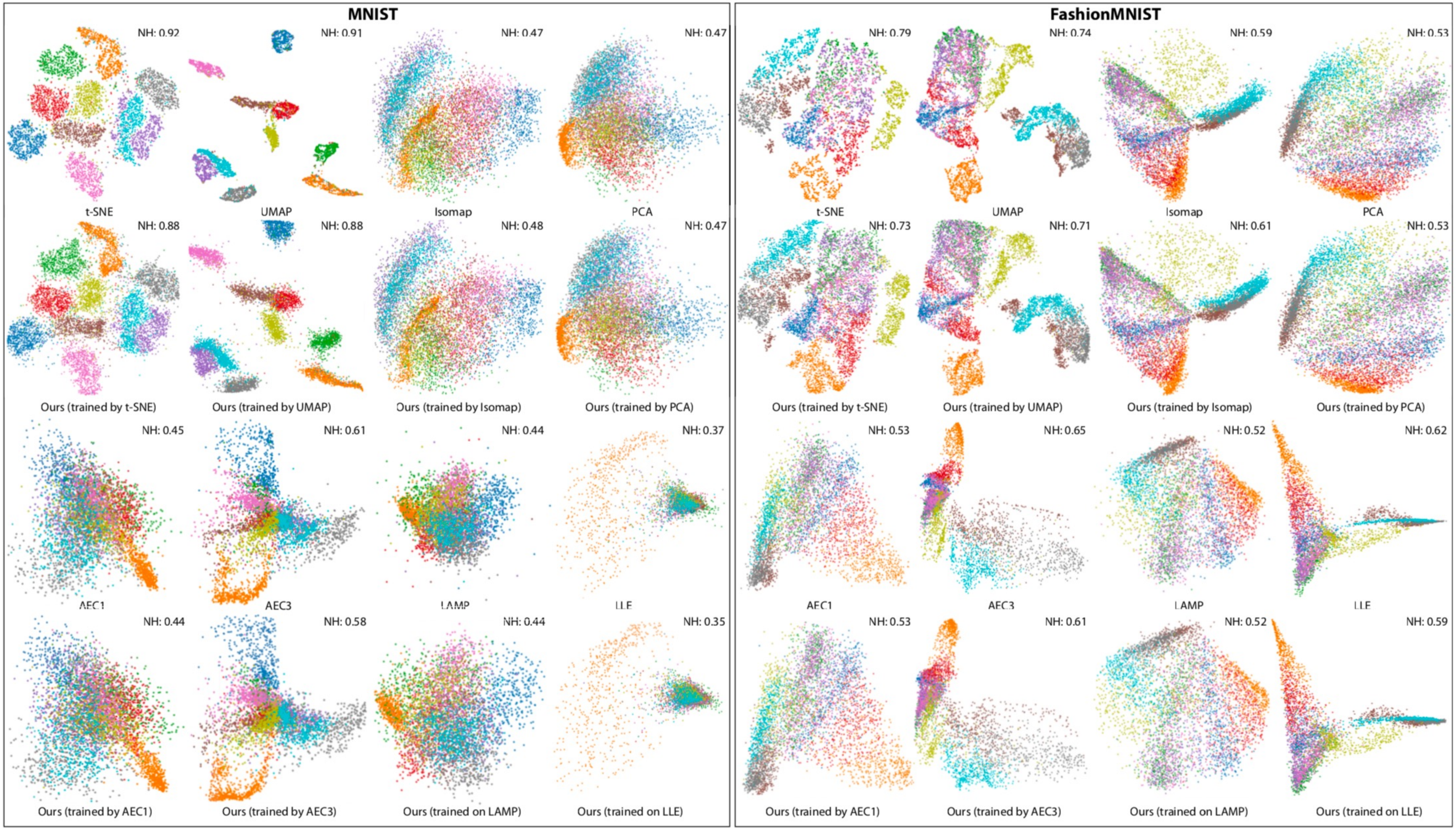
Espadoto *et al.* (2020) Deep learning multidimensional projections. *Inf Vis* 9(3):247–269

Learning different projection styles

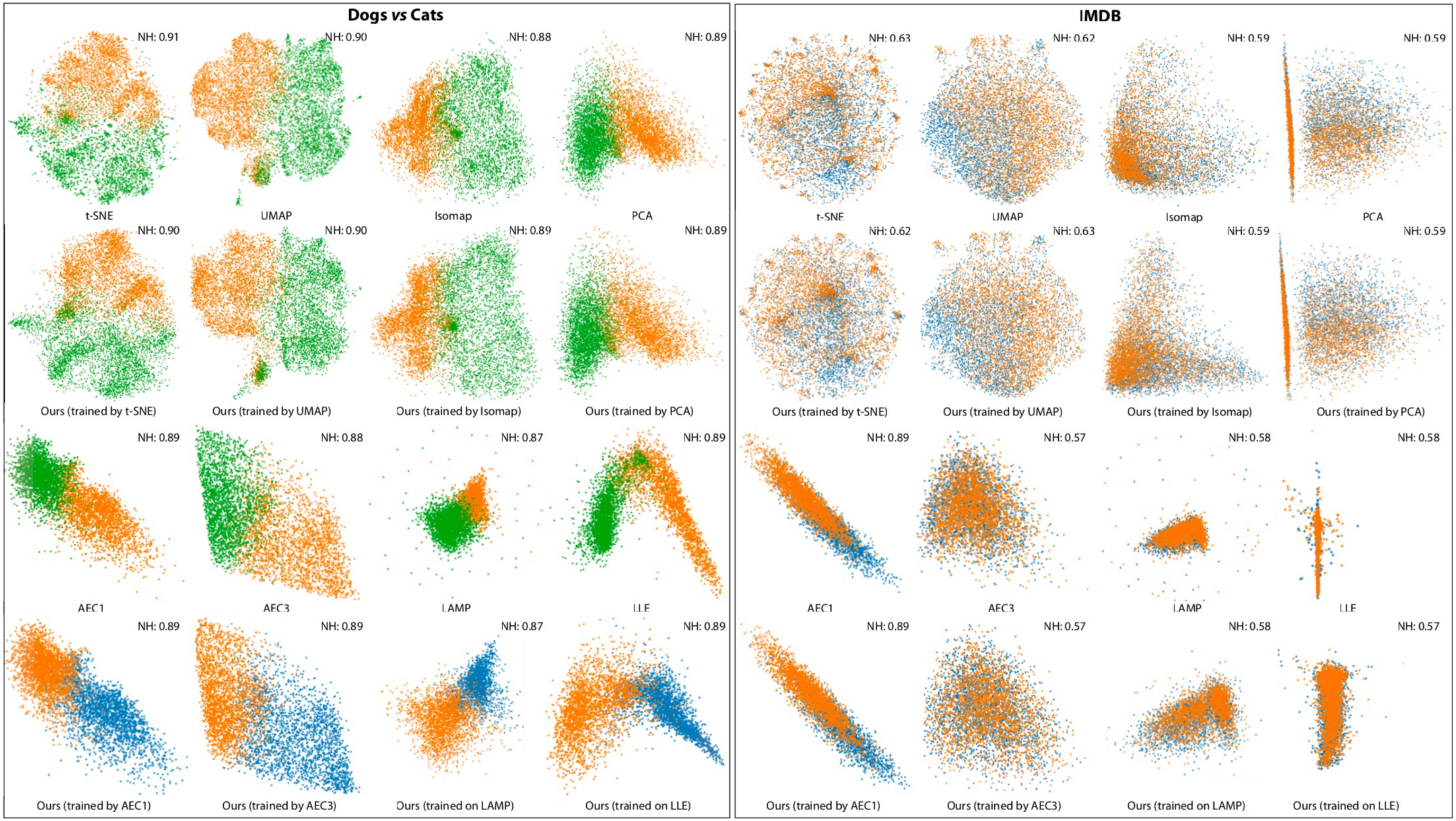


- we can imitate basically **any style**
- but, of course, the output quality will depend on the training material's quality (good `professor' = good quality, and conversely ☺)

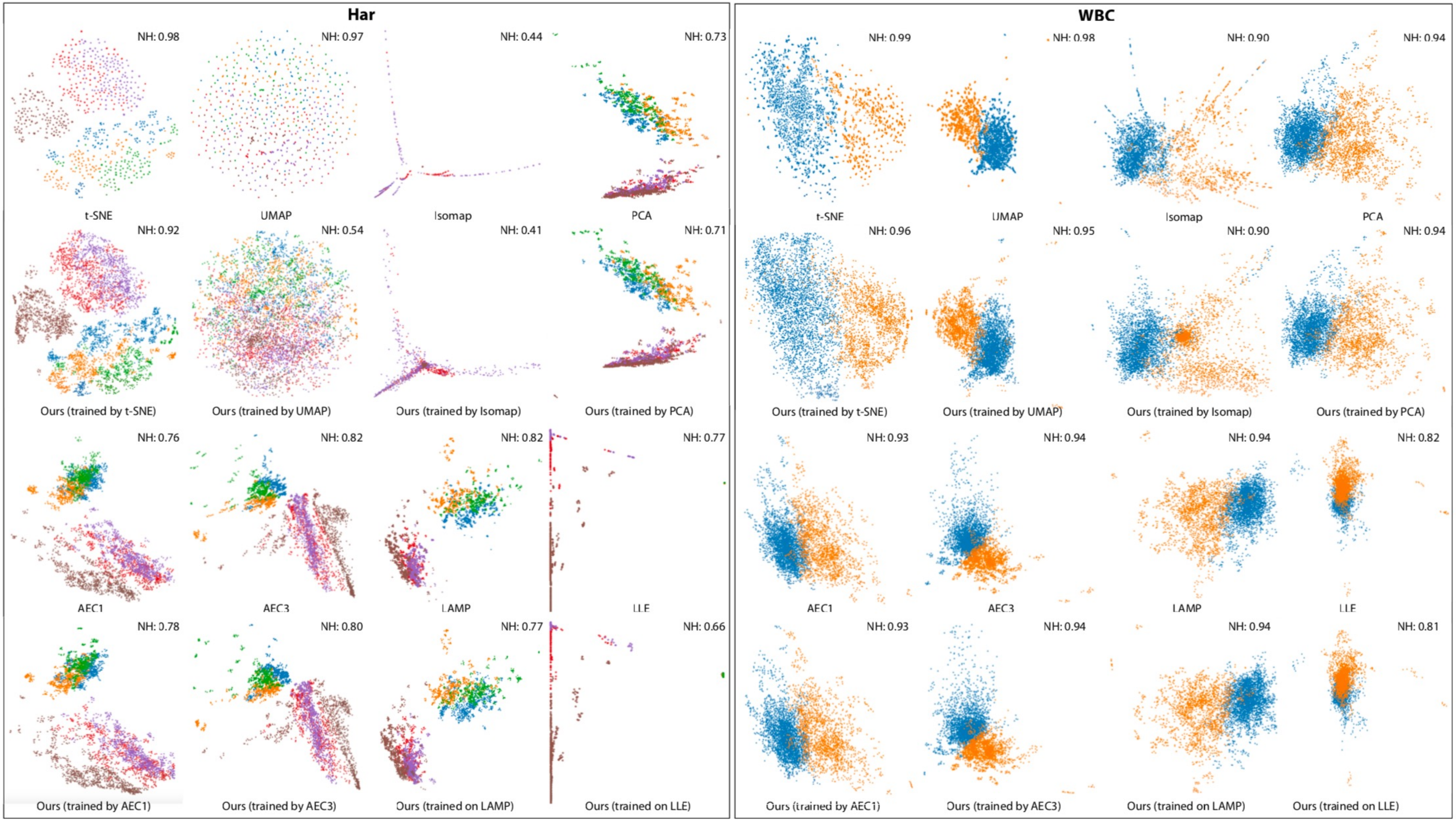
Learning different projection styles (cont'd)



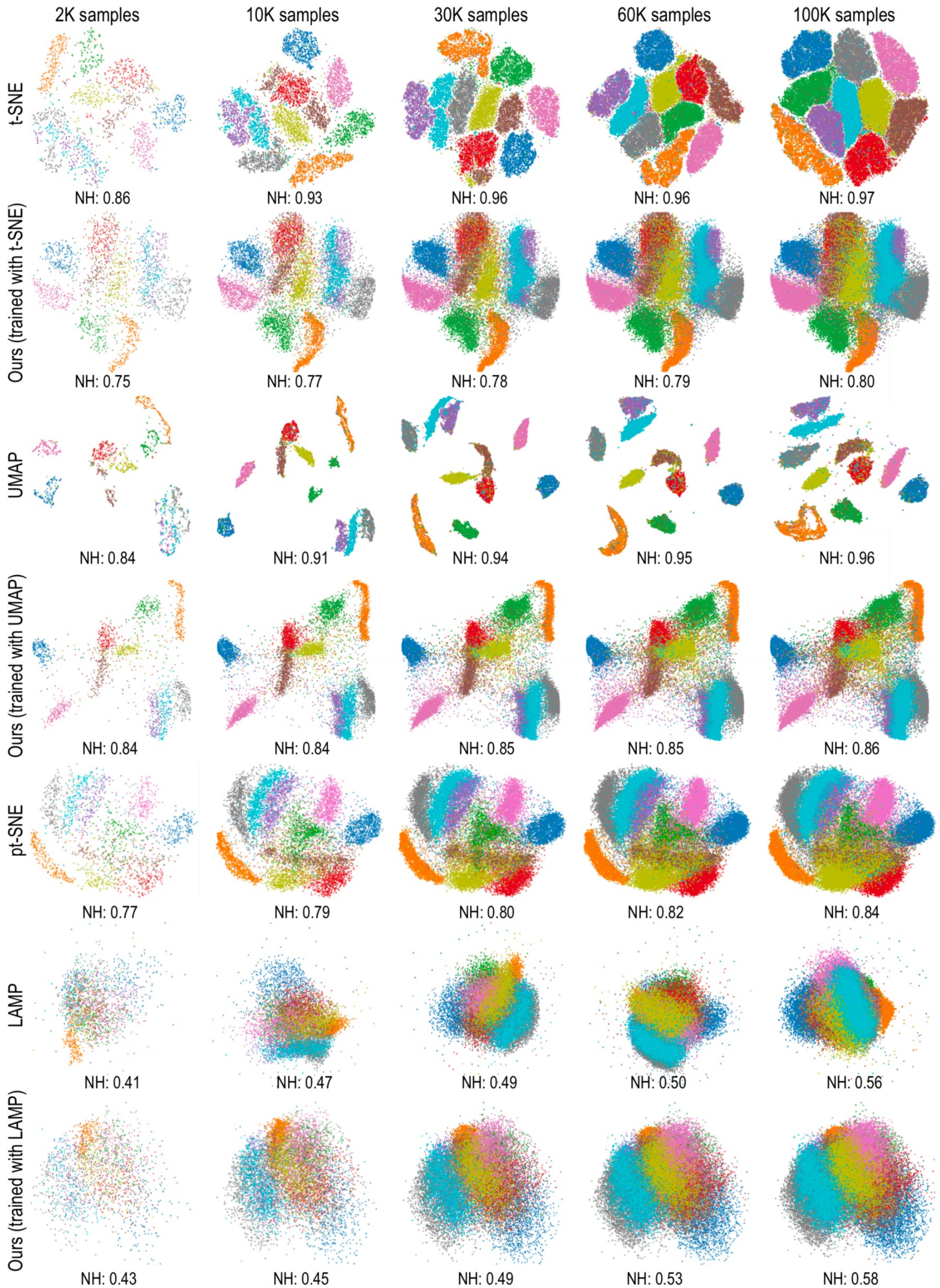
Learning different projection styles (cont'd)



Learning different projection styles (cont'd)



Out of sample capability



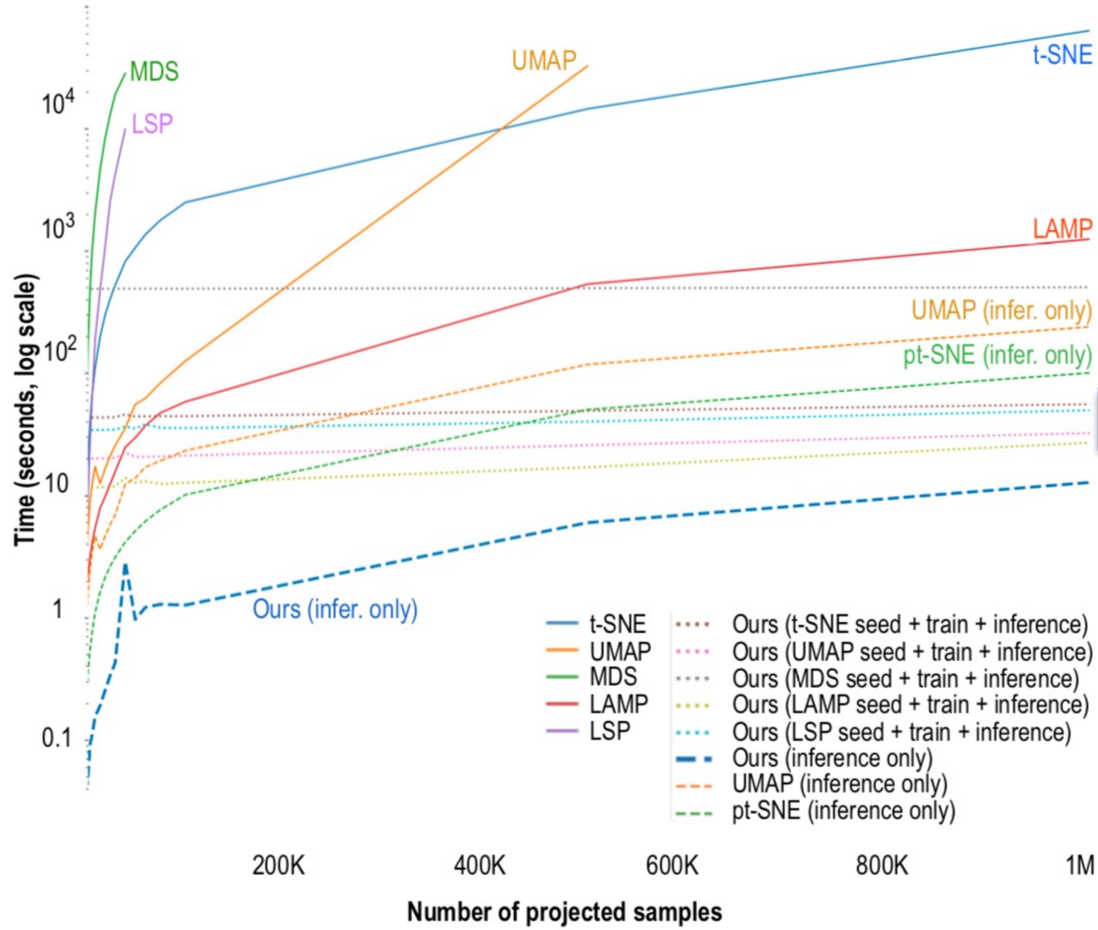
Testing

- train on a dataset D_0
- add samples to D_0 to create D_1, D_2, ...D_n
- project $P(D_0)$,...,$P(D_n)$
- compare with ground-truth $P^g(D_0)$,... $P^g(D_n)$

Results

- our method is always **stable** (out-of-sample capability by construction)
- most other methods are **not**
- we are close to the quality of parametric t-SNE (pt-SNE)

Computational scalability



ours (seed + train + inference)

Training + inference costs

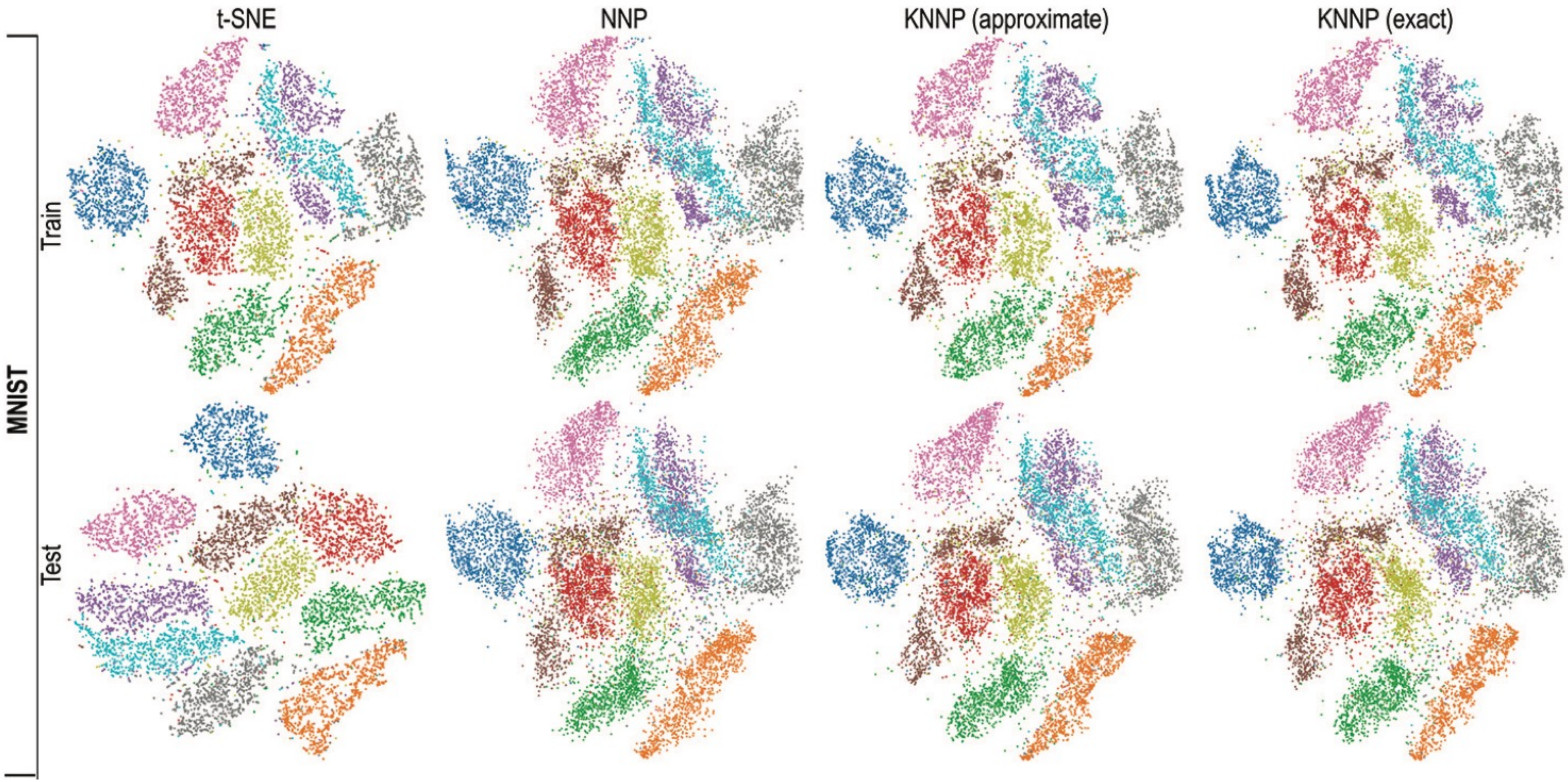
- **3K faster** than t-SNE, 2K faster than LAMP
- UMAP, LSP, MDS failed handling 1M points

Inference-only costs

- **3.5K faster** than t-SNE, 2K faster than LAMP
- 10x faster than pt-SNE

Code freely available: https://github.com/mespadoto/dlmp

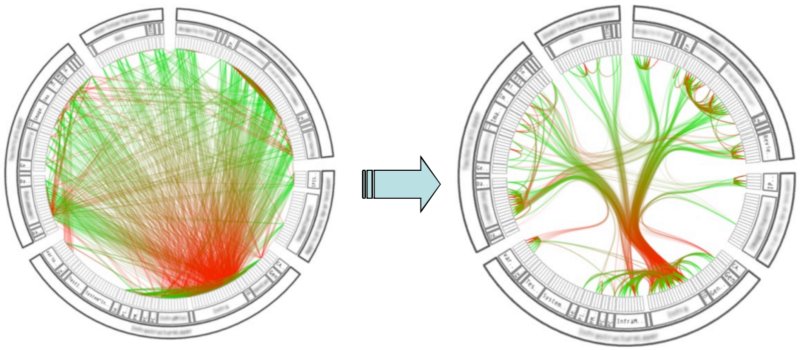
kNNP: Improve NNP by training on *neighborhoods*



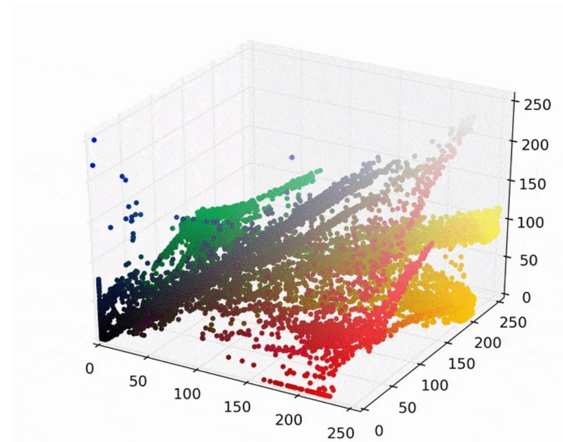
ground truth | fuzzy | less fuzzy | even less fuzzy

T. Modrakowski *et al.* (2021) Improving Deep Learning Projections by Neighborhood Analysis. Springer CCIS

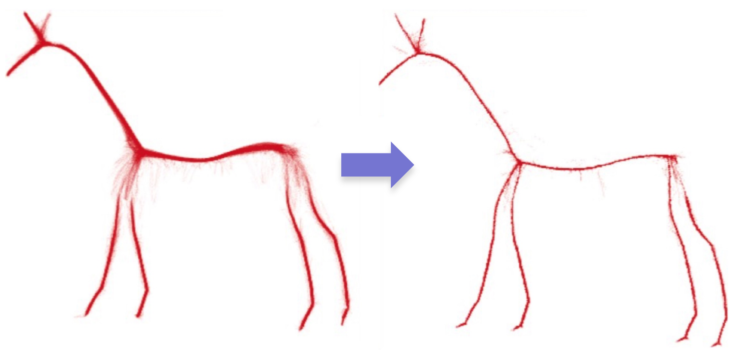
Intermezzo: Sharpening Data by Mean Shift (MS)



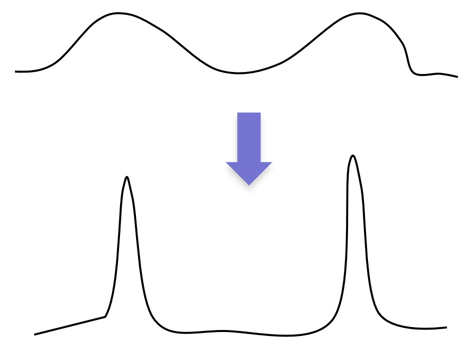
graph bundling



data clustering



point cloud processing



input signal
(fuzzy)

output signal
(sharp)

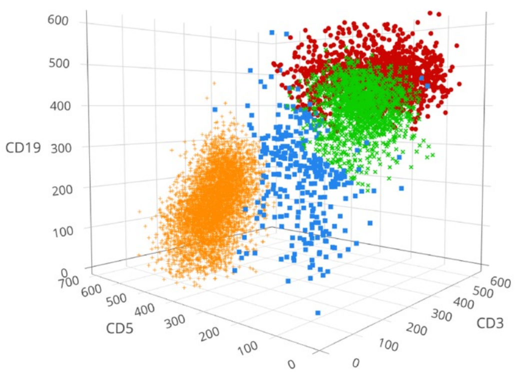
general idea

Mean shift is a very powerful, generic, tool for **finding clusters** in a data distribution!

D. Comaniciu, P. Meer (2002) Mean shift: a robust approach toward feature space analysis. IEEE TPAMI 24(5), 603-619

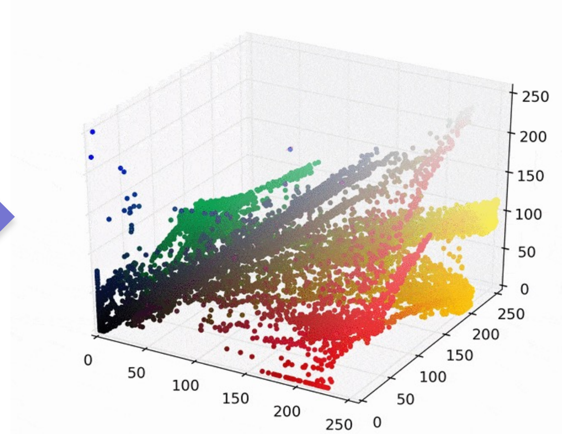
Mean Shift for Projections

can be low-quality



input nD dataset

MS



P

sharpened nD dataset

projection with clear clusters

Proposed idea

must be high-quality!

P

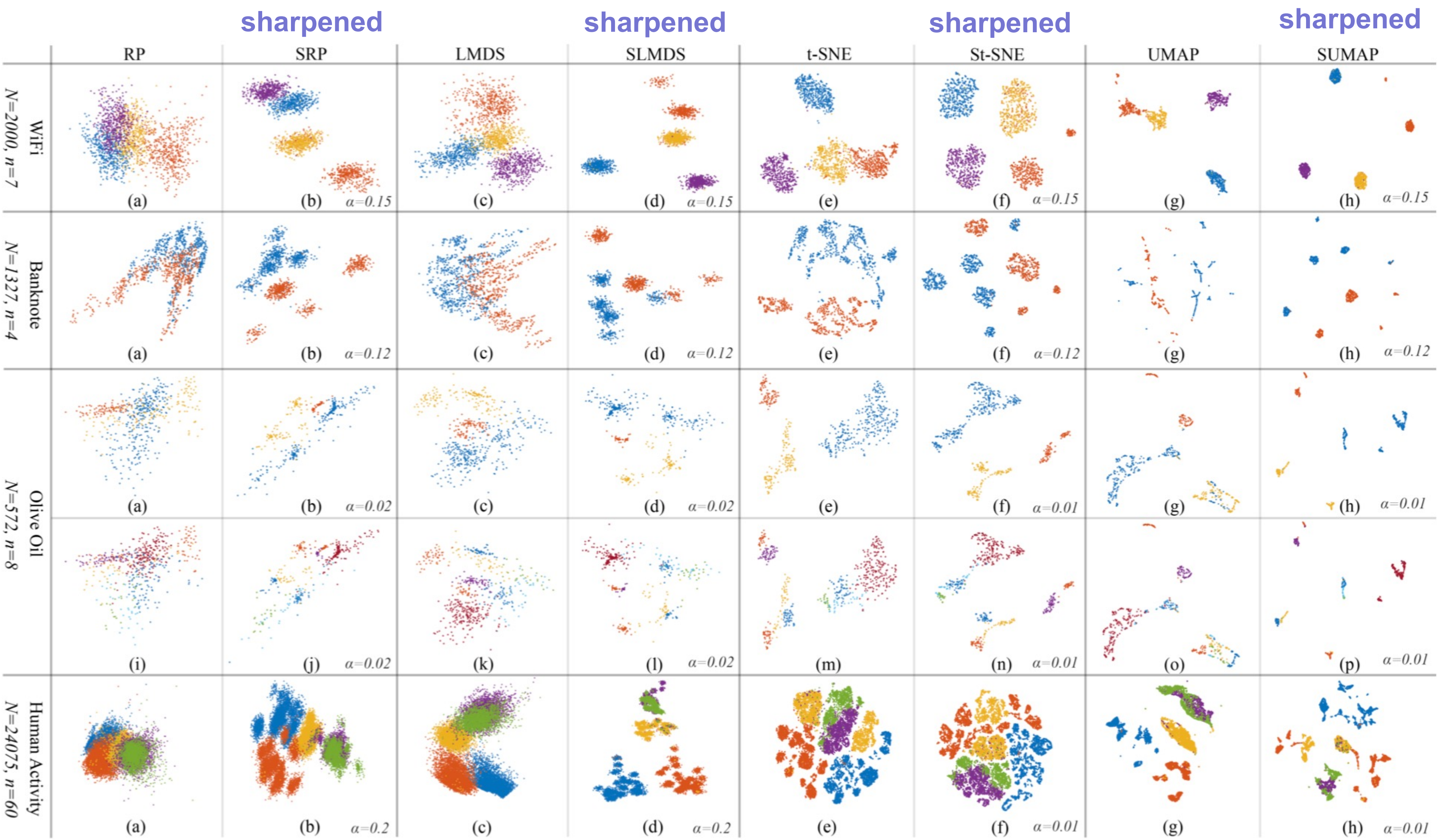
projection with clear clusters

Key idea

- use MS to **sharpen** the nD data
- clusters are then clearer **in nD** already
- so you can project this sharpened data by **simpler, faster, hackier** DR methods and still get good results

Kim *et al*. (2022) Visual Cluster Separation Using High-Dimensional Sharpened Dimensionality Reduction. Inf Vis 21(3)

Mean Shift for Projections

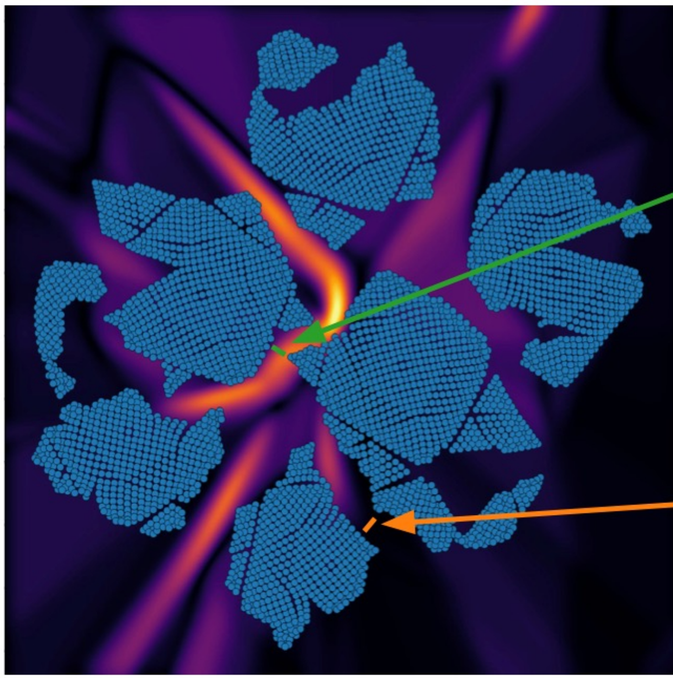


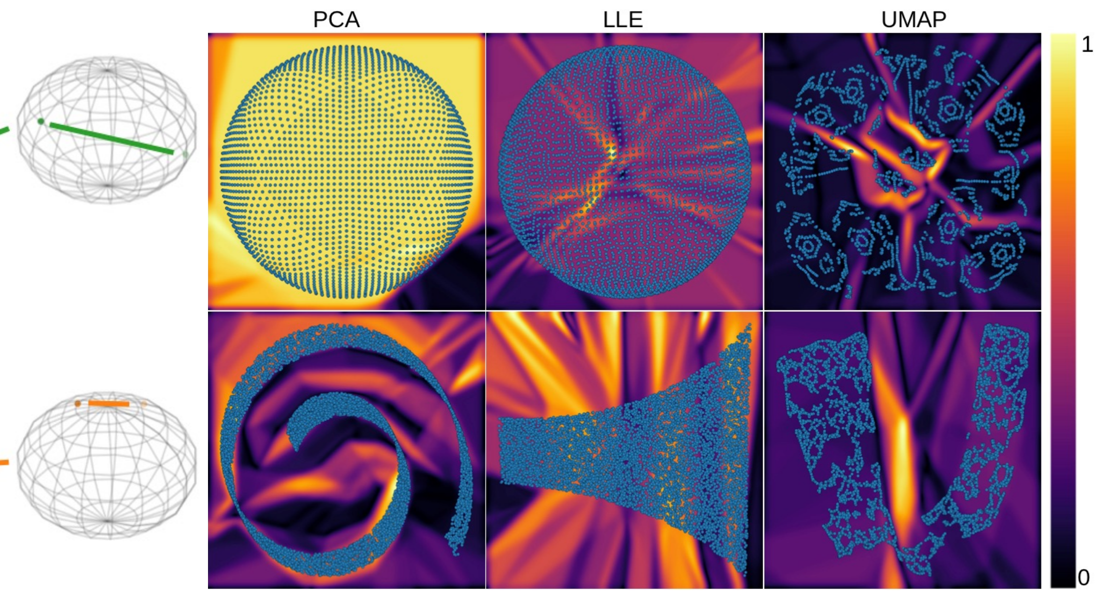
- for DR methods which **cannot** easily separate clusters well, sharpening brings **a lot** (LMDS, RP)
- for methods which are already sharp, we don't gain much (t-SNE, UMAP)
- this is exactly the idea: make **simpler/faster DR methods deliver better**!

The Way Forward

ML for DR: How to measure the quality of inverse projections?

Gradient maps: Show areas where not to trust the (inverse) projection





warm colors: *tears* in projection

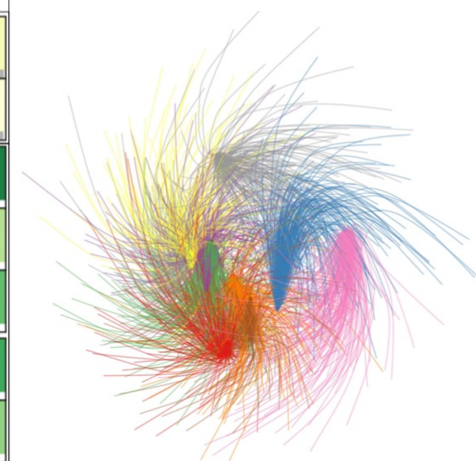
A general framework for inverse-projection errors is missing!

Espadoto *et al*. (2021) UnProjection: Leveraging inverse-projections for visual analytics of high-dimensional data. IEEE TVCG

ML for DR: How to project time-dependent data?

Methods		Distance preservation				Neighborhood preservation				Temporal stability			
	Metrics	$S_{Pearson}$	$S_{Spearman}$	$S_{Kendall}$	S_{Stress}	S_{NH}	S_{NP}	S_{Trust}	S_{Cont}	$T_{Pearson}$	$T_{Spearman}$	$T_{Kendall}$	T_{Stress}
Autoencoders	AE	0.740	0.804	0.659	0.519	0.588	0.497	0.907	0.879	0.486	0.672	0.564	1.026
Autoencoders	VAE	0.760	0.803	0.659	0.479	0.583	0.493	0.895	0.875	0.549	0.685	0.581	0.900
Graph/neighborhood methods	TF-t-SNE	0.477	0.577	0.442	1.045	0.592	0.573	0.921	0.902	0.020	0.002	0.002	1.959
Graph/neighborhood methods	G-t-SNE	0.660	0.704	0.531	0.679	0.549	0.432	0.816	0.808	0.329	0.487	0.386	1.340
Graph/neighborhood methods	dt-SNE	0.609	0.675	0.514	0.781	0.479	0.408	0.808	0.797	0.184	0.192	0.147	1.631
Graph/neighborhood methods	TF-UMAP	0.497	0.617	0.472	1.005	0.572	0.542	0.907	0.884	0.119	0.089	0.060	1.760
Graph/neighborhood methods	G-UMAP	0.610	0.672	0.502	0.779	0.508	0.387	0.771	0.779	0.264	0.316	0.234	1.471
PCA variants	TF-PCA	0.784	0.810	0.669	0.431	0.544	0.493	0.917	0.874	0.312	0.453	0.354	1.374
PCA variants	G-PCA	0.778	0.805	0.665	0.442	0.546	0.485	0.904	0.867	0.586	0.673	0.580	0.827

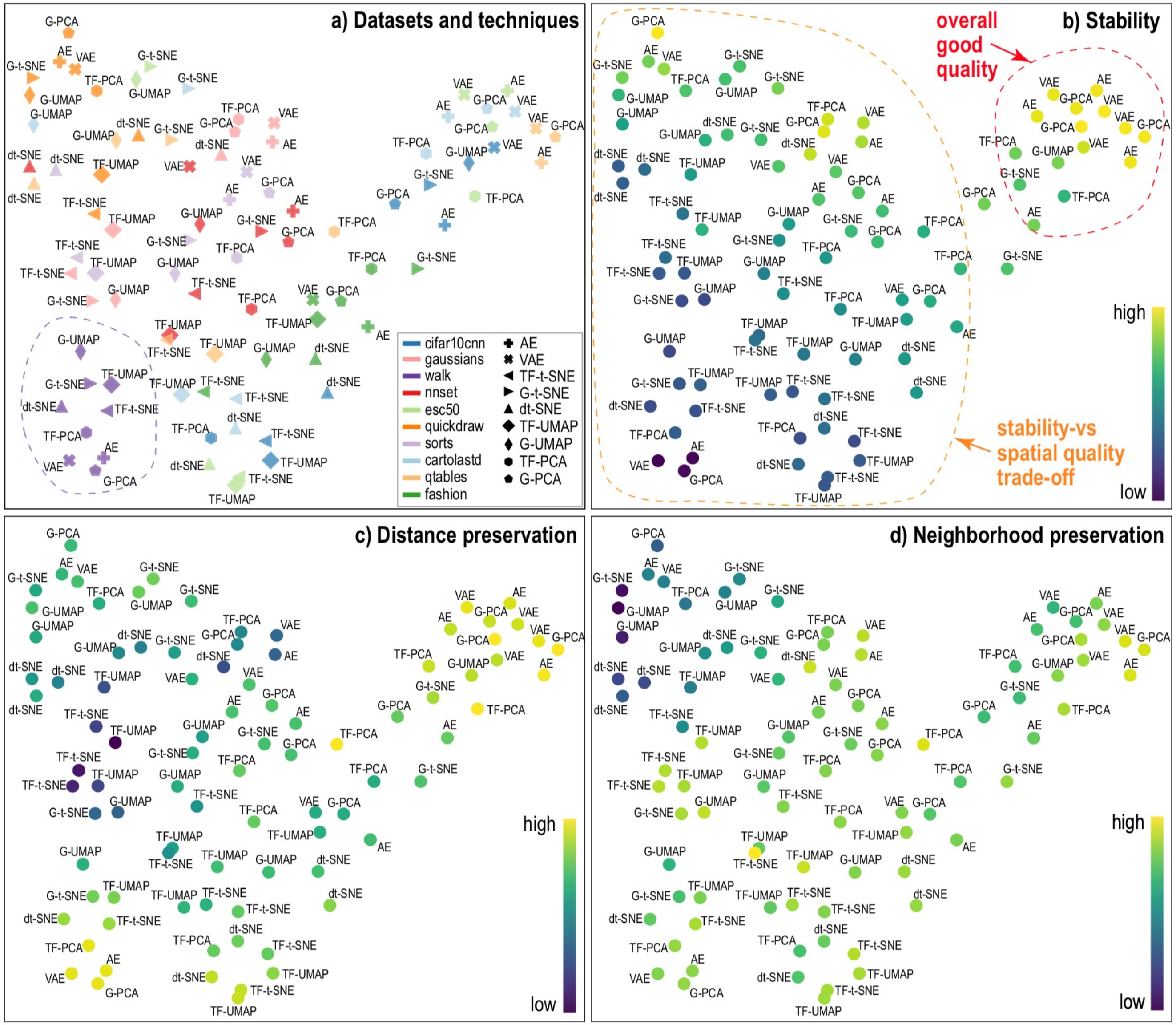
low quality — high quality



Only a handful of dynamic projection techniques exist
Visual quality and stability seem to be mutually exclusive goals (!)

E. Vernier *et al.* (2020) Quantitative Evaluation of Time-Dependent Multidimensional Projection Techniques. CGF

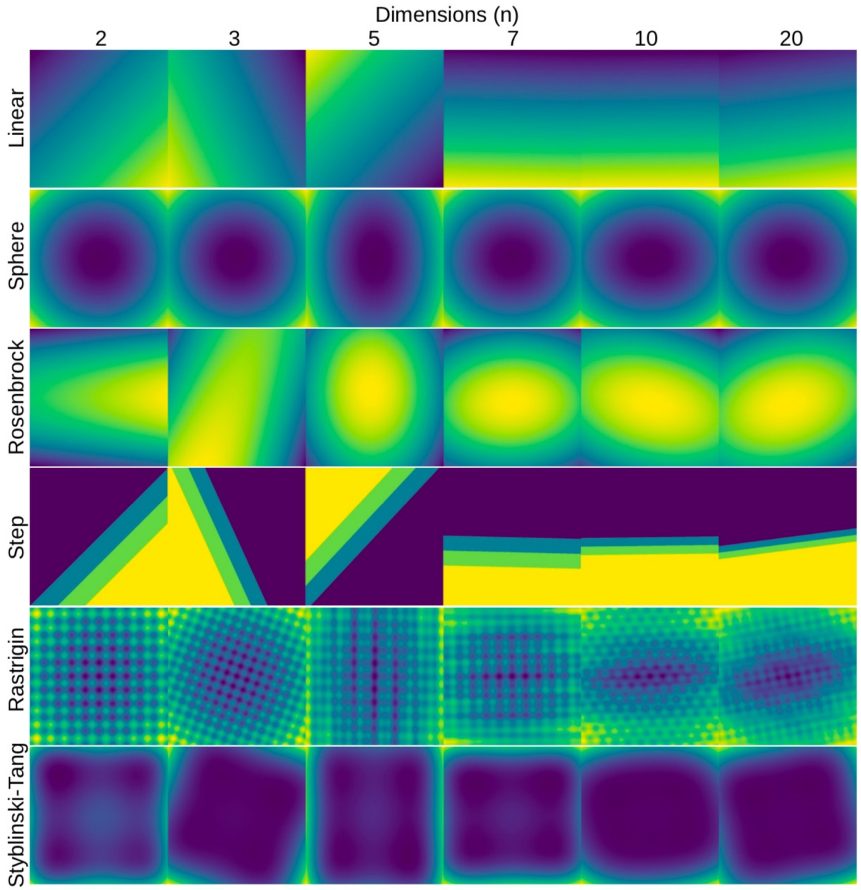
ML for DR: How to project time-dependent data?



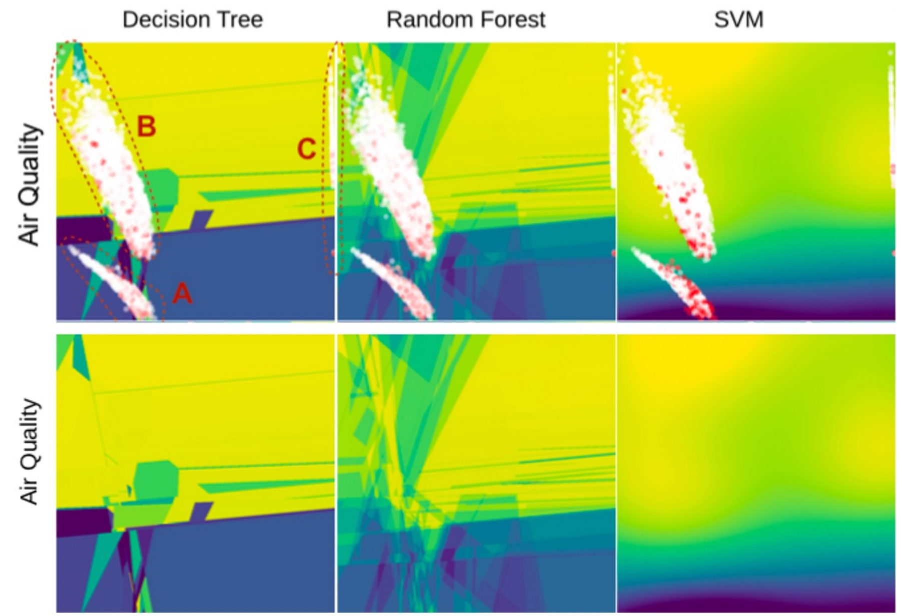
Visual quality and stability seem to be mutually exclusive goals (!)

E. Vernier *et al.* (2020) Quantitative Evaluation of Time-Dependent Multidimensional Projection Techniques. CGF

DR for ML: Visualizing general regressors



visualizing **optimizers**



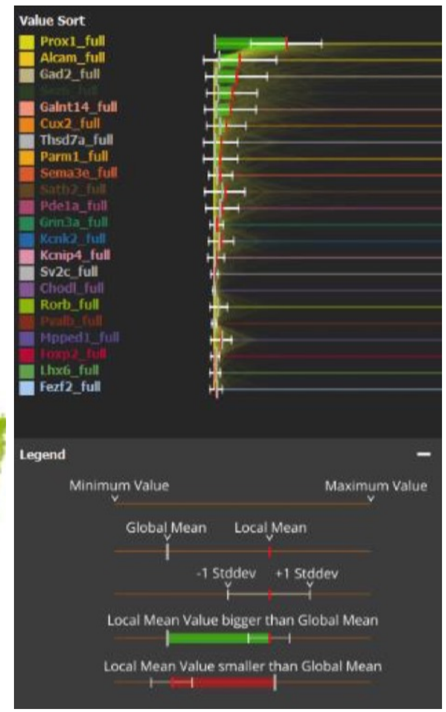
visualizing **nd-to-1D regressors**

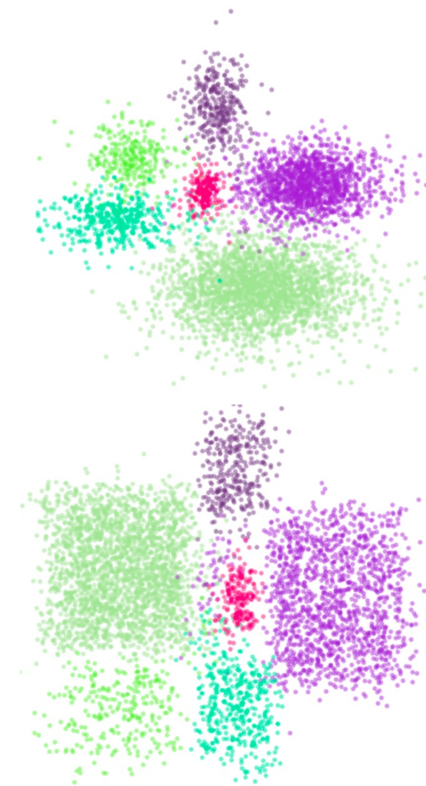
How to visualyze *any* nD-to-mD regressor?

Espadoto *et al.* (2021) OptMap: Using Dense Maps for Visualizing Multidimensional Optimization Problems. Proc. IVAPP

Espadoto *et al.* (2023) Visualizing High-Dimensional Functions with Dense Maps. Springer CCIS

DR for ML: More information in a projection





explain projection structures by dimension values

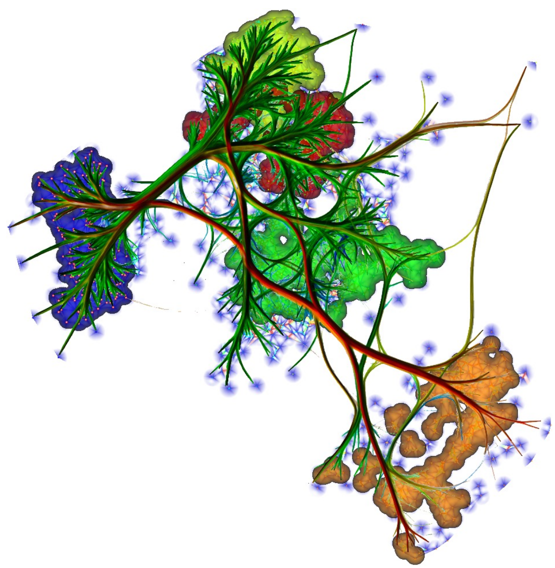
encode information in projection shape

A projection should tell a rich, complete, insightful *story* about the data!

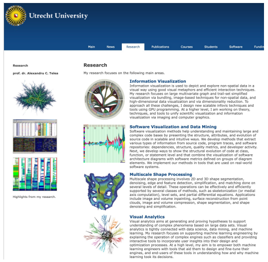
Thank you for your interest!

Alex Telea

a.c.telea@uu.nl



webspace.science.uu.nl/~telea001



- examples, applications
- code
- datasets
- papers
- people and projects

vig.science.uu.nl